Utilisation de l'intelligence artificielle pour l'aide au diagnostic des patients atteints de pathologies neuro dégénératives

Edouard Villain

Université Fédérale
Toulouse Midi-Pyrénées

THÈSE

En vue de l'obtention du
DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'Université Toulouse 3 - Paul Sabatier

Présentée et soutenue par

Edouard VILLAIN

Le 13 décembre 2021

Utilisation de l'intelligence artificielle pour l'aide au diagnostic des patients atteints de pathologies neuro dégénératives

---

Ecole doctorale : **GEETS - Génie Electrique Electronique,Télécommunications et Santé : du système au nanosystème**

Spécialité : **Radiophysique et Imagerie Médicales**

Unité de recherche :
ToNIC-Toulouse NeuroImaging Center (UMR 1214)

Thèse dirigée par
Xavier FRANCERIES et Marie-Véronique LE LANN

Jury

M. Adrian Basarab, Rapporteur
M. Olivier Saut, Rapporteur
M. Lyamine Hedjazi, Examinateur
Mme Isabelle Berry, Examinatrice
M. Xavier Franceries, Directeur de thèse
Mme Marie-Véronique Le Lann, Co-directrice de thèse

Utilisation de l'intelligence artificielle pour l'aide au diagnostic des patients atteints de pathologies neuro dégénératives

Manuscrit de thèse

Edouard VILLAIN

Direction de la thèse :
Dr. Xavier Franceries
Pr. Marie-Véronique Le Lann

Examinateurs :
Pr. Isabelle Berry
Dr. Lyamine Hedjazi

Rapporteurs de la thèse :
Pr. Adrian Basarab
Dr. Olivier Saut

Invité :
Dr. Patrice Péran

LAAS CNRS - Inserm ToNIC
Université Toulouse III - Paul Sabatier
Toulouse, France
13 Décembre 2021

Résumé

L'intelligence artificielle connait un boom depuis les années 2000 de par le stockage systématique des données et l'augmentation de la puissance de calcul des ordinateurs ainsi que l'apparition des méthodes dites de deep learning. Cela a permis d'envisager des recherches et applications dans de nombreux domaines, et en particulier le domaine médical.

Les pathologies neurodégénératives sont des fléaux pour la société depuis leurs apparitions plus fréquentes suite à l'augmentation de l'espérance de vie. Leurs diagnostics deviennent un enjeu majeur de la neuro imagerie, en particulier sur les stades précoces des pathologies. En effet, un diagnostic précoce permet d'appliquer au plus vite un traitement et de limiter les conséquences pour les patients, ainsi que de mieux comprendre les mécanismes de développement de ces pathologies et leur apparition.

Le développement d'un pipeline de deep learning appliqué aux pathologies neurodégénératives permettrait d'envisager des outils d'aide au diagnostic dans les routines cliniques, basés sur les méthodes d'intelligence artificielle les plus avancées.

Ces travaux de thèse montrent qu'il est possible d'utiliser un pipeline d'apprentissage profond à la fois pour discriminer les patients pathologiques des sujets sains, mais aussi pour effectuer une analyse des pouvoirs discriminants des biomarqueurs dérivés de l'IRM. Ils démontrent comment obtenir une signature spatiale de la pathologie étudiée, tout en utilisant un jeu de données compatible avec une routine clinique. La pathologie neurodégénérative étudiée est l'atrophie multi systématisée, maladie rare avec peu donnée de patients, pour laquelle le deep learning parvient néanmoins à un bon diagnostic.

Ces travaux pourraient être étendus à d'autres pathologies dégénératives, à la fois pour le diagnostic et le suivi des patients, mais aussi pour la compréhension de ces pathologies.

Abstract

Artificial intelligence has experienced a boom since the 2000s due to the systematic storage of data and the increase in the computing power of computers as well as the emergence of so-called deep learning methods. This made it possible to envisage research and applications in many fields and in particular the medical field.

Neurodegenerative pathologies have plagued society since their more frequent appearances following increased life expectancy. Their diagnoses are becoming a major issue in neuroimaging, in particular in the early stages of pathologies. Indeed, an early diagnosis makes it possible to apply treatment as quickly as possible and to limit the consequences for patients, as well as to better understand the mechanisms of development of these pathologies and their appearance.

The development of a deep learning pipeline applied to neurodegenerative pathologies would make it possible to consider diagnostic aid tools in clinical routines, based on the most advanced artificial intelligence methods.

This thesis work shows that it is possible to use a deep learning pipeline both to discriminate pathologic patients from healthy subjects, but also to perform an analysis of the discriminating powers of biomarkers derived from MRI. They demonstrate how to obtain a spatial signature of the pathology studied, while using a data set compatible with a clinical routine. The neurodegenerative pathology studied is multisystem atrophy, a rare disease with little patient data, for which deep learning nevertheless reaches a good diagnosis.

This work could be extended to other degenerative pathologies, both for the diagnosis and the follow-up of patients, but also for the understanding of these pathologies.

Remerciements

Ce doctorat a été une étape très importante de ma vie, durant laquelle j'ai pris plaisir à exploiter mes compétences dans le domaine de l'intelligence artificielle en les appliquant au milieu passionnant de la neuro imagerie.
Cela m'a permis d'apprendre énormément sur le domaine de la recherche mais aussi de l'enseignement.
Tout cela n'aurait pu être possible sans le Dr. Xavier Franceries et la Pr. Marie-Véronique Le Lann, mes directeurs de thèse, que je tiens à remercier chaleureusement.
Ces travaux n'auraient pu aboutir sans la collaboration avec le Dr. Patrice Péran, le Dr. Federico Nemmi ainsi que Giulia Maria Mattia.
Je tenais aussi à remercier la Pr. Isabelle Berry qui m'a accueilli dans son équipe d'enseignement de biophysique pour le PACES, qui a accepté d'être la présidente du jury de thèse, mais aussi pour sa confiance dans le développement d'un sous module d'intelligence artificielle appliquée à l'imagerie médicale dans son Master Imagerie Médicale.
Je remercie aussi le Pr. Denis Kouamé et le Pr. Adrian Basarab, mes encadrants de stage de Master, qui m'ont donné le goût de la recherche et m'ont donné envie de poursuivre en Doctorat.
Je remercie aussi le Directeur de recherche Olivier Saut et le Pr. Adrian Basarab d'avoir accepté de rapporter la thèse ainsi que le Dr. Lyamine Hedjazi d'avoir accepté d'être examinateur de la thèse.
Je remercie bien sûr les laboratoires du LAAS CNRS et de l'inserm ToNIC pour leurs accueils, ainsi que l'école Doctorale GEET.
Je n'oublie pas non plus de remercier l'Université de Toulouse Paul Sabatier et ses enseignants. Je pense notamment au Dr. Thomas Pellegrini et au Dr. Jerôme Farinas qui m'ont enseignés les bases de l'intelligence artificielle indispensables à la réussite de ce Doctorat.
Je remercie aussi mes amis futurs docteurs Thomas Rolland, Hugo Rens et Frederic Chatrie avec qui j'ai pu échanger tout au long du doctorat.
Pour finir je tiens à remercier ma famille, mes parents et ma soeur, qui ont toujours été là pour moi, que ce soit pour fêter les réussites mais aussi me soutenir durant les moments de doutes.
Afin de n'oublier des personnes, je tiens à remercier tous ceux et toutes celles qui auraient pu participer à l'élaboration de cette thèse.

Edouard Villain

Table des matières

Introduction

L'intelligence artificielle (IA) a pour but d'automatiser un traitement en tentant de reproduire le résultat que pourrait obtenir un humain. Les applications de l'intelligence artificielle sont très nombreuses, que ce soient les outils d'aide au diagnostic, la traduction et la reconnaissance du langage, le traitement du signal, les finances, la robotique ou encore le domaine médical.

La réduction du coût du stockage des données a entrainé une sauvegarde systématique de la plupart des données. Couplée à l'augmentation de la puissance de calcul des machines, l'intelligence artificielle connaît un boom depuis les années 2000.

Ces travaux de thèse intitulés "Utilisation de l'intelligence artificielle pour l'aide au diagnostic des patients atteints de pathologies neurodégénératives" sont au carrefour de plusieurs disciplines. L'intelligence artificielle avec les modèles d'apprentissage profond (dit deep learning), l'aide au diagnostic et le domaine médical avec la neuro imagerie.

En effet, en partant de l'imagerie par résonance magnétique (IRM) et les biomarqueurs dérivés, l'objectif est de construire des outils d'aide au diagnostic basés sur le deep learning 3-dimensions multimodal utilisable en routine clinique. L'atrophie multi systématisée (AMS), un syndrome Parkinsonien rare et agressif, sert d'exemple d'application.

Ces travaux de thèse tentent de répondre à plusieurs questions :

— Le deep learning permet-il d'obtenir une discrimination entre les patients atteints de pathologies neurodégénératives et les sujets sains en utilisant un jeu de données de l'ordre de la dizaine de patients ?

— Le deep learning permet-il d'analyser le pouvoir discriminant des biomarqueurs dérivés de l'IRM ?

— Le deep learning permet-il de définir une signature spatiale de la pathologie étudiée ?

Ce manuscrit de thèse est alors composé d'un chapitre sur l'état de l'art sur l'intelligence artificielle, la neuro imagerie et l'application de l'IA à la neuro imagerie.
Le chapitre suivant détaille les techniques et méthodologies permettant de développer un réseau de neurones.

Ensuite un chapitre présente les différentes méthodes de visualisation des zones discriminantes d'un réseau de neurones ainsi que la méthode développée durant ces travaux de thèse.
Dans le but de valider la méthode de visualisation ainsi que les modèles de réseau de neurones développés, le chapitre 4 utilise des images de biomarqueurs dérivés de l'IRM simulées pour étudier la capacité des modèles à discriminer une population de sujets sains et de sujet comprenant une anormalité dans l'image. La visualisation des zones discriminantes est aussi appliquée afin de confirmer que la méthode développée durant ces travaux de thèse est capable de retrouver la zone ou l'anormalité a été simulée.

Après avoir montré que la capacité d'un réseau de neurones à discriminer deux populations sur des biomarqueurs dérivés de l'IRM, ainsi que la capacité à retrouver les zones du cerveau incriminées, le pipeline construit durant cette thèse est ensuite appliquée à l'AMS dans le chapitre 5 avec un jeu de données de petite taille compatible avec une application clinique. Il est composé d'une trentaine de patients AMS et d'une trentaine de sujets sains, avec plusieurs biomarqueurs dérivés de l'IRM par patient et sujet. Les résultats de la discrimination entre les deux populations sont alors présentés pour plusieurs types d'architectures de réseaux de neurones. L'analyse du pouvoir discriminant des biomarqueurs ainsi que la visualisation des zones discriminantes complètent l'application du pipeline d'aide au diagnostic des pathologies neurodégénératives.
Les résultats de l'application à l'AMS du pipeline basé sur le deep learning sont comparés avec ceux obtenus par F. Nemmi et al. dans une étude publiée [46]. Le manuscrit de thèse se termine par une conclusion et des perspectives.

CHAPITRE 1

État de l'art

1.1 État de l'art de l'Intelligence Artificielle

L'intelligence artificielle regroupe différents types de paradigmes et d'algorithmes. Ses objectifs sont toujours d'automatiser un traitement ou encore calculer des prédictions. Certains algorithmes sont dits supervisés lorsqu'il est nécessaire de fournir les labels (valeurs recherchées ou valeurs cibles) pour l'entrainement et d'autres sont dit non-supervisés lorsque l'algorithme n'utilise aucune information a priori.

Trois grandes parties composent l'intelligence artificielle :

— Les systèmes experts
— Le machine learning
— Le deep learning

Chacune de ces parties dispose de différents avantages, inconvénients et applications. Elles sont détaillées dans les sections 1.1.1, 1.1.2 et 1.1.3.

Ces trois grandes parties sont illustrées sur la figure 1.1.

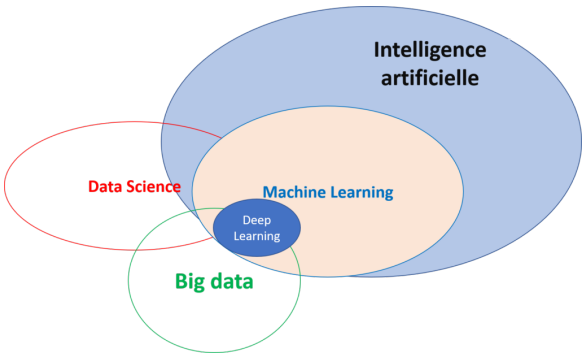


FIGURE 1.1: Principaux paradigmes d'intelligence artificielle

1.1.1 Système expert

Un système expert vise à reproduire une intelligence et s'appuie sur trois parties :
— Une base de faits
— Une base de règles
— Un moteur d'inférence

La base de faits contient une liste de faits permettant par exemple de décrire un environnement. La base de règles est composée de règles logiques à appliquer au fait. Le moteur d'inférence quant à lui utilise la base de règles en l'appliquant à la base de faits dans le but de produire de nouveaux faits. Plus d'informations sur les systèmes experts peuvent être trouvées dans [1, 2].

L'exemple suivant permet d'illustrer un système expert.

Soit F une base de faits permettant de représenter un environnement, composée des faits p, q. Soit R une base de règles contenant les formules logiques suivantes :
$p \wedge q \vdash r$
$p \wedge r \vdash s$

A l'initialisation, $F : \{p, q\}$.
En appliquant le moteur d'inférence, une première itération permet de déduire le fait r via la règle $p \wedge q \vdash r$, et F devient $F : \{p, q, r\}$.
Une seconde itération déduit le fait s avec la règle $p \wedge r \vdash s$ et F devient $F : \{p, q, r, s\}$.

Afin de rendre plus parlant cet exemple, supposons que les faits correspondent au cas suivant :
— p : il y a du soleil
— q : le taux d'humidité est faible
— r : il ne pleut pas
— s : le sol est sec

Un robot a alors la possibilité d'utiliser ses capteurs de lumière et d'humidité pour détecter les faits p et q. En saturant la base de faits via l'application de la base de règles, le robot peut déduire qu'il ne pleut pas et que le sol est sec ce qui lui permet d'adapter sa motricité en fonction de son environnement.

La difficulté des systèmes experts est située dans la rédaction des règles logiques. De plus, les temps de calcul peuvent être longs en fonction de la taille de la base de règles puisqu'elles vont toutes être appliquées plusieurs fois jusqu'à ce que plus aucun fait ne soit produit.

1.1.2 Machine learning

Le machine learning rassemble les algorithmes de statistiques automatiques permettant de calculer une prédiction telle qu'une classification, une segmentation ou une régression de données. Plusieurs algorithmes sont regroupés sous le terme de machine learning et les plus connus sont les K plus proches voisins (K-nn), les machines à vecteur de support (SVM) ou encore les K-moyennes (k-means) et les fôrets aléatoires [3].

Les K plus proches voisins est un algorithme de classification qui utilise une métrique de distance et s'appuie sur les labels des K voisins d'une nouvelle donnée afin de lui attribuer une classe [4, 5]. La figure 1.2 montre un exemple de classification d'une donnée par l'algorithme des K plus proches voisins.

Les K-moyennes est un algorithme de classification non-supervisé qui cherche les K centroïdes des classes de façon à minimiser la distance moyenne des données d'une classe à son centroïde.

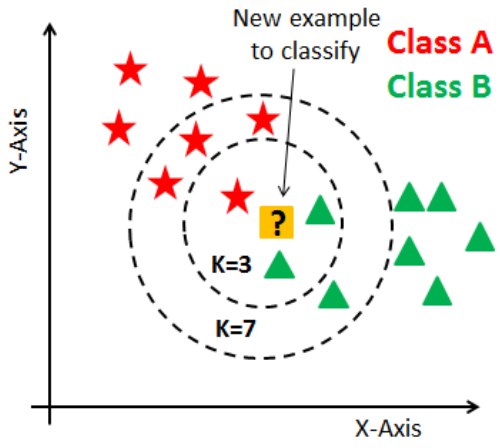


FIGURE 1.2: Algorithme des K plus proches voisins (K-nn)

Les algorithmes de K plus proches voisins et K-moyenne nécessitent que l'utilisateur définisse K, le nombre de voisins pour les K-nn ou de classes pour les k-means.

Les machines à support de vecteur permettent de calculer une classification en maximisant la distance entre les classes. Pour cela, une fonction noyau peut être appliquée afin d'augmenter la dimension des données dans le but de trouver un espace dans lequel une classification linéaire est possible [6, 7, 8, 9]. La figure 1.3 détaille un exemple de classification par SVM avec la marge maximum entre les deux classes et l'hyperplan optimal les séparant.

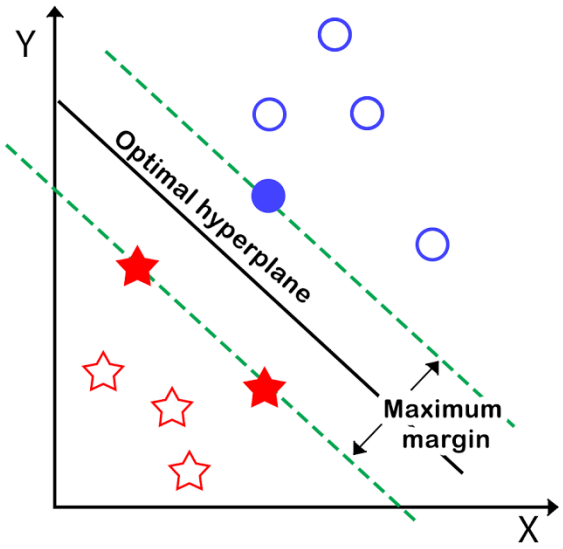


FIGURE 1.3: Algorithme de SVM

1.1.3 Deep learning

Le deep learning regroupe les algorithmes d'apprentissage profond avec les réseaux de neurones capables de traiter de grande base de données (Big data) pour des applications de classification, de segmentation et de régression.
Le chapitre suivant détaille le fonctionnement d'un neurone artificiel (voir la section 2.1) et du réseau de neurones (voir la section 2.2).

Ce neurone artificiel est apparu en 1943 avec les travaux de Warren Mcculloch et Walter Pitts [10] permettant d'effectuer des calculs logiques. Cependant, ce modèle du neurone artificiel ne peut apprendre à calculer une discrimination non linéaire. Ces travaux sur le neurone artificiel sont repris en 1958 avec le perceptron développé par Rosenblatt [11].

En 1982 les premiers réseaux de neurones apparaissent avec les travaux de Hopfield [12]. Les travaux de Yann LeCun permettent d'améliorer l'apprentissage du neurone artificiel avec l'algorithme de rétro propagation en 1985 [13], suivi par l'amélioration de cet algorithme en 1988 proposée par Rumelhart, Hinton et Williams [14].

Yann LeCun propose une adaptation de la rétro propagation en 1989 permettant de développer les réseaux de neurones convolutifs [15]. Son modèle permet de classifier les images contenant des chiffres manuscrits.

Hochreiter et Schmidhuber ont proposé un type de réseau neuronal récurrent avec mémoire à court et long terme en 1997 [16].

Suite à ces travaux initiant les réseaux de neurones artificiels, de nombreuses études ont été réalisées afin de diversifier les tâches de ces modèles, mais aussi leurs architectures. La compétition annuelle image-net a permis de développer plusieurs architectures [17] et les modèles entrainés sont disponibles dans la littérature afin d'effectuer du transfer learning (voir la section 2.4). Il est alors possible de citer les modèles VGG [18], ResNet [19] et Googlenet [20] (voir la section 2.3).

1.2 État de l'art de la neuroimagerie

1.2.1 Imagerie par résonance magnétique

L'imagerie par résonance magnétique (IRM) fait partie des examens cliniques d'imagerie à disposition des praticiens médicaux. Tout comme pour les échographies et les scanners, l'objectif est d'obtenir une image du corps humain.

L'IRM utilise les champs intenses via une bobine à supraconducteur refroidie à l'hélium liquide ce qui permet d'éviter les effets de chauffe et donc les effets indésirables pour le corps humain. Afin d'obtenir une image, la première étape consiste à déterminer les noyaux des éléments à étudier. En effet, seuls les noyaux avec un spin non nul sont sensibles aux effets de résonance magnétique.
Ces noyaux sont ensuite mis en précession a une vitesse angulaire $\omega = \gamma \times B_0$ avec γ le rapport gyromagnétique de l'élément étudié et B_0 l'intensité du champ électro magnétique. Ceci correspond aussi à la fréquence de Larmor $\nu = \frac{\gamma}{2\pi} B_0$.
Dans le cas des noyaux avec un spin de $\frac{1}{2}$, deux états énergétiques sont alors possibles avec une population parallèle $N_{\parallel}$ et une population anti-parallèle $N_{\nparallel}$ avec $E = \pm\frac{1}{2}\gamma\frac{h}{2\pi}B_0$. La différence éner-

gétique entre ces deux niveaux est alors notée ΔE.

Le noyau de l'élément étudié entre alors en précession si $\Delta E = h\nu$ avec ν égal à la fréquence de Larmor.

Une impulsion radio fréquence est ensuite appliquée avec une intensité B_1 pour une durée t. Cette impulsion permet de faire basculer l'angle de l'aimantation macroscopique $\vec{M_0}$ d'un angle $\varphi = \gamma B_1 \Delta t$. Deux composantes apparaissent alors, avec la composante transversale et longitudinale, respectivement égale à :
$M_x = M_0 \sin \varphi$
$M_z = M_0 \cos \varphi$

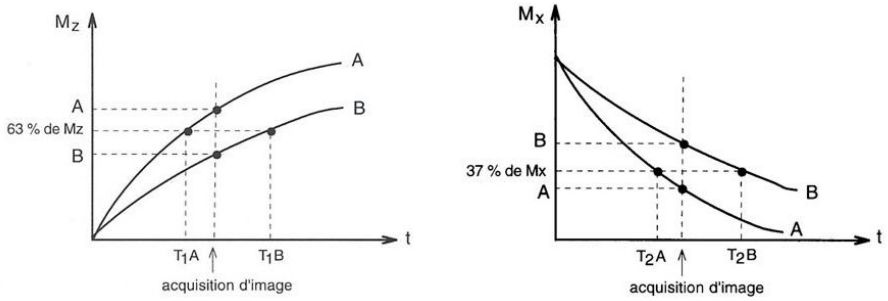


FIGURE 1.4: Temps de relaxation T1 et T2

Une acquisition d'une image IRM étudie les valeurs des composantes transversales M_x et longitudinales M_z à un instant t suivant l'impulsion radio fréquence B_1 comme le montre la figure 1.4. Les images pondérées en T_1 correspondent à l'étude de la composante longitudinale M_z tandis que les images pondérées en T_2 utilisent la composante transversale M_z.

La figure 1.5 montre une machine IRM.

FIGURE 1.5: Dispositif d'imagerie par résonance magnétique

1.2.2 Biomarqueurs dérivés

L'acquisition d'une image IRM fournit des données brutes et de grandes dimensions. Les biomarqueurs dérivés de l'IRM permettent de mettre en évidence certaines parties de l'information contenue dans les données brutes de l'IRM, ou encore de mettre en évidence certains phénomènes. Les principaux biomarqueurs dérivés de l'IRM utilisés en *neuro imagerie* sont listés ci-dessous.

- GM (Gray Matter) : permet de mettre en évidence le niveau de substance grise et est utile en neuro imagerie car le volume de substance grise à tendance à réduire ou s'affaisser chez les patients atteints de pathologies neurodégénératives.
- MD (Mean Diffusivity) : image le mouvement Brownien des molécules d'eau. Ce biomarqueur est utile pour le diagnostic des pathologies neurodégénératives car les patients atteints de ce type de pathologies ont souvent des agrégats ferreux qui modifient le mouvement Brownien des molécules d'eau [21].
- ALFF (Amplitude of Low Frequency Fluctuation) : permet de représenter l'activité du cerveau en imageant le niveau d'oxygénation dans le sang dans une séquence temporelle. Seules les basses fréquences sont conservées car les hautes fréquences correspondent aux battements du coeur, or l'objectif de ALFF est d'obtenir les micro-variations du niveau d'oxygénation du sang correspondant à l'activité cérébrale. Finalement l'amplitude est conservée ce qui permet d'obtenir une image de l'activité cérébrale plutôt qu'une séquence temporelle.
- FA (Fractional Anisotropy) : permet de mesurer l'anisotropie de la diffusion. Une FA de 0 correspond à une diffusion isotropique, autrement dit qu'aucune direction n'est priviligiée, alors qu'une FA de 1 correspond à une diffusion contrainte selon une seule direction [22].
- R2* : permet d'imager le dépôt de fer.
- Local Correlation : permet de mesurer la cohérence locale d'un voxel par rapport à son voisinage.
- Global Correlation : permet de mesurer la corrélation entre un voxel et tous les autres voxels représentant le cerveau.

La table 1.1 regroupe toutes les informations sur les biomarqueurs dérivés de l'IRM avec le type d'imagerie, l'acquisition ainsi que les logiciels utilisés pour les calculer.

Biomarqueur	Type d'image	Acquisition	Toolbox de calcul
GM	Structurelle	Images T1	CAT12 [23]
MD	Diffusion	DWI (Diffusion Weighted Images)	FSL [24]
ALFF	Fonctionnelle	Resting State functional MRI	SPM [25]
FA	Diffusion	DWI (Diffusion Weighted Images)	FSL [24]
R2*	Fonctionnelle	Images T2*	SPM [25]
Local correlation	Fonctionnelle	Resting State functional MRI	SPM [25]
Global correlation	Fonctionnelle	Resting State functional MRI	SPM [25]

TABLE 1.1: Acquisitions et calculs des biomarqueurs dérivés de l'IRM du cerveau

1.2.3 Template MNI

Les templates MNI sont des atlas développés par le Montreal Neurological Institute qui ont pour but de résoudre les problèmes de localisation dans les images IRM [26]. En effet chaque patient étant unique, les acquisitions d'IRM fournissent des images où les cerveaux ont des tailles différentes et les régions du cerveau peuvent être décalées d'un patient à l'autre. Cela devient problématique lorsqu'une localisation est nécessaire.

Afin de palier ce problème l'Institut Neurologique de Montreal a proposé ces templates MNI (ou atlas) en moyennant les cerveaux de centaines de patients afin d'obtenir un cerveau moyen. Ensuite, une transformation non linéaire permet de ramener une nouvelle acquisition vers un template MNI. Il existe plusieurs templates MNI disponibles dans la littérature permettant d'obtenir des tailles physiques de voxels allant de $0.5 \times 0.5 \times 0.5mm^3$ à $3 \times 3 \times 3mm^3$ [27, 28].

1.2.4 Parkinson

La maladie de Parkinson est une maladie neurodégénérative qui se développe généralement entre 45 et 70 ans. Elle est la seconde maladie la plus fréquente après la maladie d'Alzheimer.
La maladie de Parkinson se distingue des syndromes Parkinsoniens qui sont souvent plus agressifs et répondent moins aux traitements.

Le diagnostic de ces pathologies neurodégénératives est difficile car tous les syndromes Parkinsoniens et la maladie de Parkinson ont en commun plusieurs symptômes.

— Le tremblement des extrémités
— Une rigidité des mouvements
— Des mouvements lents

Si certains syndromes Parkinsoniens peuvent être diagnostiqués, la maladie de Parkinson reste à ce jour une maladie idiopathique.

Parmi ces syndromes Parkinsoniens, il est possible de citer :

— L'atrophie multi systématisée (AMS)
— La paralysie supra nucléaire (PSP)
— La dégénérescence corticobasale (DCB)
— La maladie à corps de Lewy

Ces syndromes Parkinsoniens sont souvent confondus avec la maladie de Parkinson de par les symptômes en commun. L'évolution de la pathologie est souvent l'un des éléments permettant de discriminer les syndromes Parkinsoniens de la maladie de Parkinson.

Ces syndromes Parkinsoniens sont des pathologies rares et agressives, répondant peu aux traitements. Leur étude est un challenge car la collecte de données de pathologies rares, au diagnostic définitif, est difficile.
Plus d'informations sur la maladie de Parkinson peuvent être trouvées dans [29].

1.3 Application de l'Intelligence Artificielle à la neuro imagerie

L'intelligence artificielle appliquée au domaine du médical est un sujet de recherche rassemblant une très grande communauté, proposant des études et solutions pour tous les types d'applications médicales et utilisant les trois grandes parties de l'intelligence artificielle, que ce soit les systèmes experts tels que MYCIN [30], le machine learning [31] et le deep learning [32].

Cependant dans le domaine de la neuro imagerie, les méthodes de machine et deep learning sont les plus représentées. Les applications sont variées allant de la segmentation d'image [33, 34, 35], la génération d'images via les modèles Generative Adversial Network (GAN) [36], la régression telle que la prédiction de l'âge d'un sujet en s'appuyant sur une IRM cérébrale [37] ainsi que la classification visant à discriminer une population de sujets sains et une population de patients pathologiques [38].

Toutes les pathologies neurologiques peuvent être étudiées via l'intelligence artificielle que ce soit la maladie d'Alzheimer [39, 40, 41, 42, 43], la maladie de Parkinson [38, 44] et les syndromes Parkinsoniens [45, 46, 47], la neuro fibromatose [48] ou encore la sclérose [49].
D'autres travaux utilisent l'intelligence artificielle et la neuro imagerie pour des sujets de recherche plus vastes, et moins souvent étudiés tels que l'étude du coma [50], les troubles neurologiques et psychiatriques [51] ou encore la différence entre les cerveaux d'un homme et d'une femme [52].
Certains travaux ajoutent une partie visualisation et interprétation des prédictions de leur modèle dans le but d'obtenir une signature spatiale de la pathologie étudiée. F. Nemmi et al. proposent une approche basée sur les SVM qui fournit la discrimination entre les syndromes Parkinsonien et les sujets sains ainsi que les clusters de voxels les plus discriminants correspondant à une signature spatiale des pathologies [46]. Esmaeilzadeh et al. proposent aussi une signature spatiale de la maladie de Parkinson en 2018 avec un modèle de réseau de neurones convolutifs 3-dimensions [32]. Chengliang et al. utilisent aussi le deep learning et l'interprétation visuelle pour obtenir une signature spatiale de la maladie d'Alzheimer [53].

Plusieurs recherches se concentrent sur la création d'une plateforme d'intelligence artificielle appliquée à la neuro imagerie en proposant des solutions à tous les types de problèmes (segmentation, régression et classification) telles que BrainNetCNN [54] et NiftyNet [55]. Ils proposent alors des architectures prêtes à l'emploi ou encore des modèles déjà entrainés.

Parmis ces travaux de recherches, les objectifs peuvent être variés. P. Péran et al. comparent les différents algorithmes de classification supervisés et non-supervisés en discriminant le même jeu de données sur les syndromes Parkinsoniens [47]. Nemmi et al. ont développé un pipeline de classification appliqué aux syndromes Parkinsoniens [46], en validant leurs résultats via les connaissances présentes dans l'état de l'art de la pathologie [56, 57, 31] et en les comparant avec les résultats d'autres méthodes publiées [58, 59, 60, 61, 62, 63, 47, 64]. Ce pipeline est ensuite appliqué à d'autres pathologies [65].

S'il existe de nombreux travaux d'application de l'intelligence artificielle à la neuro imagerie, très peu se concentrent sur les pathologies neuro dégénératives rares où la collecte de données est difficile. La taille des jeux de données des pathologies rares est limitée ce qui rend difficile l'utilisation des techniques d'intelligence artificielles dites profondes.
De plus, peu de travaux visent à développer un pipeline entièrement basé sur le "deep learning" en combinant IRM 3-dimensions, biomarqueurs dérivés de l'IRM et multimodalité dans le but d'obtenir à la fois une classification, mais aussi une analyse du pouvoir discriminant des différentes biomarqueurs ainsi qu'une visualisation spatiale des zones discriminantes. Ce type de pipeline avec peu de données permet pourtant d'étudier les pathologies rares en utilisant les techniques d'intelligence artificielles les plus avancées, mais aussi une application clinique via les données collectées sur le centre clinique.

CHAPITRE 2

Développement d'un modèle de réseau de neurones artificiels

Le chapitre 3 permet de détailler l'apprentissage profond, la méthode utilisée lors de cette thèse.

Dans un premier temps, le neurone artificiel et son apprentissage seront abordés en section 2.1. Puis les réseaux de neurones ainsi que leurs architectures seront détaillés dans les sections 2.2 et 2.3 avec leurs avantages respectifs.
Les hyper-paramètres associés aux réseaux de neurones seront expliqués dans la section 2.7.
La section 2.6 quant à elle s'intéressera à la mesure de performance d'un réseau de neurones artificiels.
Ce chapitre se terminera par les modèles mono et multi modaux dans la section 2.8 et l'entraînement d'un modèle section 2.9.

2.1 Neurone artificiel

2.1.1 Modèle du neurone artificiel

Le neurone artificiel s'appuie sur l'analogie avec le neurone biologique pour calculer une prédiction. En effet le neurone biologique peut être simplifié en trois parties fondamentales.

— Les dendrites
— Le corps cellulaire
— Les axones

Les dendrites reçoivent de l'information, le corps cellulaire quant à lui s'occupe de traiter l'information reçue et les axones de transmettre l'information traitée. Ce pipeline de traitement d'information est copié dans le neurone artificiel avec toujours ces trois parties.

— Les entrées : $[x_1, x_2, \ldots, x_n]$
— La sortie : $\sum_{i=1}^{i=n} x_i \times \omega_i + b$
— L'activation : $\varphi(\sum_{i=1}^{i=n} x_i \times \omega_i + b)$

La figure 2.1 schématise cette analogie entre le neurone biologique et le neurone artificiel.

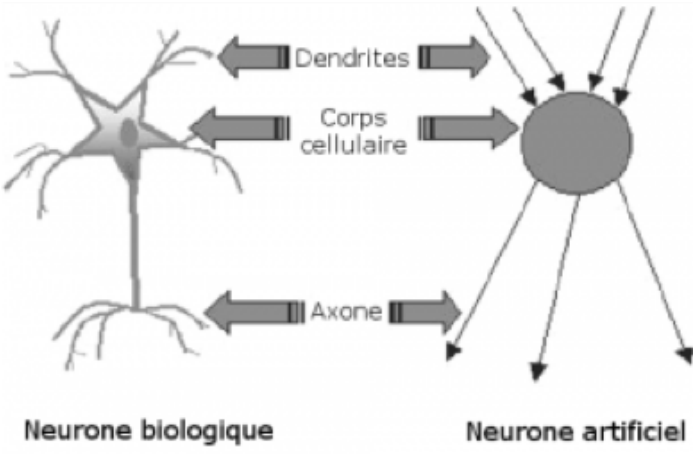


FIGURE 2.1: Analogie entre le neurone biologique et le neurone artificiel

La figure 2.2 quant à elle schématise le neurone artificiel.

A ce stade, le neurone artificiel est capable de calculer une prédiction à partir d'entrée, ou plutôt une sortie activée, grâce à un modèle mathématique simple. Cependant il n'est pas capable d'apprendre et de généraliser l'information contenue dans les données d'apprentissage.

2.1.2 Apprentissage du neurone artificiel

Afin d'obtenir la capacité d'apprendre, le neurone artificiel s'appuie sur une technique basique, correspondant à l'apprentissage de l'humain. Le cycle essais, échec et amélioration pour éviter de retomber dans un cas d'échec est répété pour minimiser le nombre d'échecs. Pour cela, il faut connaitre à l'avance ce que le neurone artificiel doit prédire dans le but de mesurer son taux d'échec. On appelle ce type d'apprentissage un apprentissage supervisé. En effet, chaque donnée possède un label associé précisant la sortie attendue pour cette donnée (appelée aussi valeur cible) : la classe d'appartenance dans le cas d'une classification, la valeur souhaitée dans le cas d'une régression. Mathématiquement, ce cycle schématisé figure 2.3, correspond aux étapes suivantes :

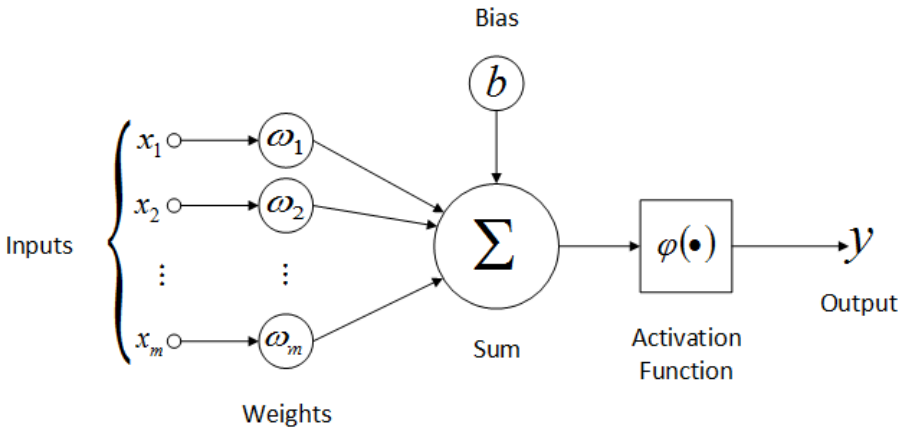


FIGURE 2.2: Modèle mathématique du neurone artificiel

— Étape 1 : Calcul de la sortie
— Étape 2 : Calcul de l'erreur entre la sortie calculée et la sortie attendue (le label)
— Étape 3 : Rétro-propagation de l'erreur en mettant à jour les poids ω_i et le biais b de façon à minimiser l'erreur de prédiction
— Étape 4 : Passage à la donnée suivante

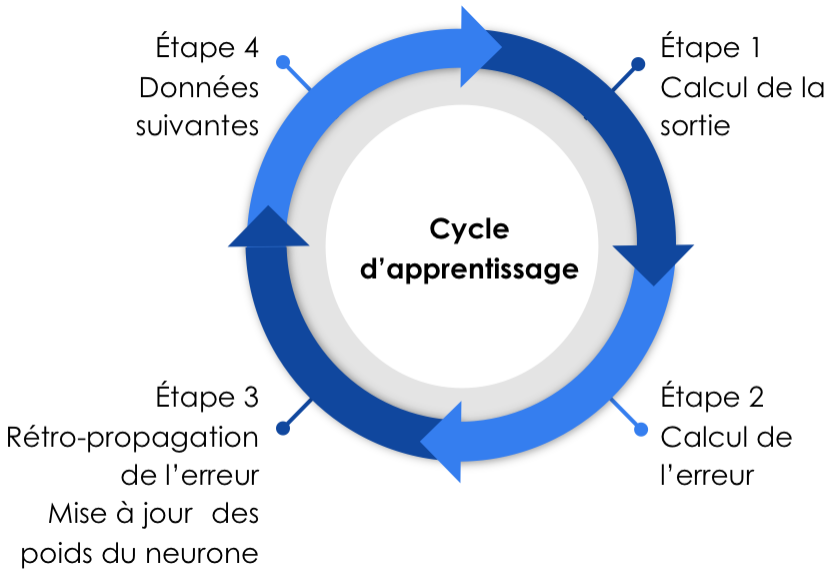


FIGURE 2.3: Cycle d'apprentissage du neurone artificiel supervisé

Ce cycle d'apprentissage est répété sur un grand nombre de données dans le but de pouvoir généraliser l'information contenue dans un jeu de données. Cela permet d'obtenir un taux d'échec

minimal sur l'ensemble des données d'apprentissage et ainsi pouvoir calculer des prédictions sur de nouvelles données. Il serait possible de schématiser de façon plus mathématique ce cycle via la figure 2.4.

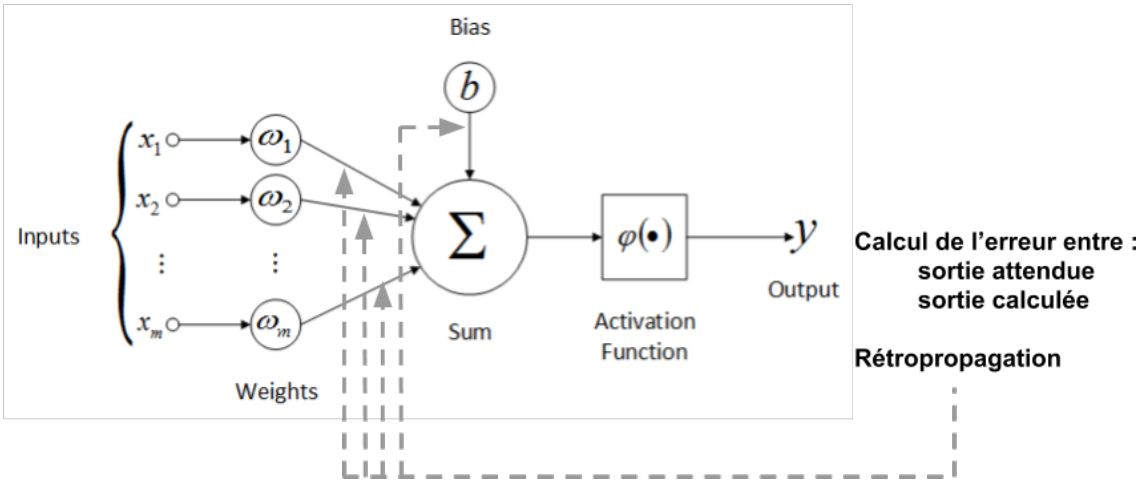


FIGURE 2.4: Apprentissage du neurone artificiel supervisé

2.1.3 Fonction d'activation du neurone artificiel

La fonction d'activation φ d'un neurone artificiel permet de réctifier la sortie calculée par l'équation $\sum_{i=1}^{i=n} x_i \times \omega_i + b$. Plusieurs types de fonctions d'activation sont disponibles. Certaines sont destinées à être intégrées à l'interieur d'un modèle et d'autres en sortie d'un modèle. Les fonctions d'activion ReLU (Rectified Linear Unit), LeakyReLU et ELU (Exponential Linear Unit) permettent d'accélérer le temps de calcul et l'apprentissage en ayant tendance à retirer les sorties négatives. La fonction ReLU met à zéro les valeurs négatives via l'équation 2.1. Le calcul de la sortie étant principalement composé de multiplications, le fait de mettre les valeurs négatives à zéro accélère énormément le temps de calcul.

$$\varphi(y) = max(0, y) \tag{2.1}$$

Les fonctions d'activation LeakyReLU et ELU quant à elles laissent passer une partie des valeurs négatives. Le neurone artificiel dispose alors de plus d'information pour calculer sa prédiction, cependant les temps de calcul sont légèrement augmentés par rapport à la fonction ReLU. L'équation 2.2 détaille le calcul de la fonction d'activation LeakyReLU.

$$\varphi(y) = \begin{cases} y \text{ si } y \geq 0 \\ \alpha \times y \text{ si } y < 0, \text{ avec } \alpha \in [0 \cdots 1] \end{cases} \tag{2.2}$$

L'inconvénient de la fonction LeakyReLU est qu'elle n'est pas dérivable en zéro. La fonction d'activation ELU dont l'équation est détaillée 2.3 cumule l'avantage de la fonction LeakyReLU tout en étant dérivable en zéro. Cependant le calcul d'une exponentielle augmente le temps de calcul,

mais améliore nettement l'apprentissage [66].

$$\varphi(y) = \begin{cases} y \text{ si } y \geq 0 \\ \alpha \times (e^y - 1) \text{ si } y < 0, \text{ avec } \alpha \in [0 \cdots 1] \end{cases} \tag{2.3}$$

Les fonctions d'activations sigmoïd, softmax et celle dite de tout ou rien sont quant à elles destinées à la sortie d'un modèle. La fonction d'activation dite de tout ou rien dont l'équation est précisée 2.4 permet une classification binaire.

$$\varphi(y) = \begin{cases} 0 \text{ si } y \leq 0 \\ 1 \text{ sinon} \end{cases} \tag{2.4}$$

La fonction softmax quant à elle permet de transformer un vecteur en probabilités, et dans le cas d'un réseau de neurones, un vecteur de probabilités d'appartenance aux classes de façon à ce que la somme des probabilités soient égales à un. L'équation 2.5 détaille le calcul utilisé avec y_i la probabilité d'appartenance à la classe i et K le nombre de classe du problème. Cette fonction d'activation est alors idéale pour la classification multi classes.

$$\varphi(y_i) = \frac{e^{y_i}}{\sum_{j=0}^{j=K} e^{y_j}} \tag{2.5}$$

La fonction sigmoïd dont le calcul est présenté équation 2.6 est utilisée pour les problèmes de régression.

$$\varphi(y) = \frac{1}{1 + e^{-y}} \tag{2.6}$$

La figure 2.5 présente les principales fonctions d'activation disponibles dans la littérature.

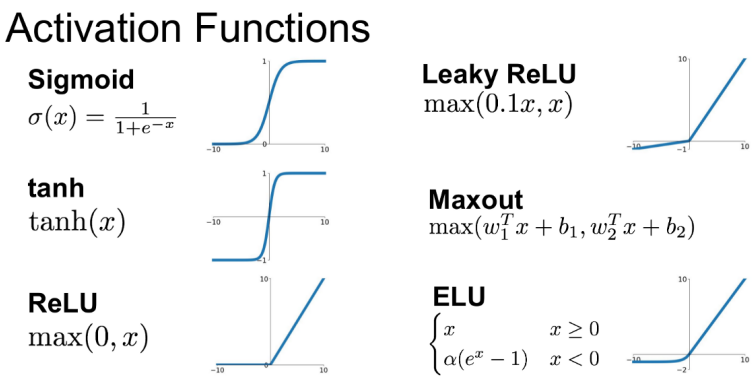


FIGURE 2.5: Principales fonctions d'activations du neurones artificiels

En poussant l'analyse du neurone artificiel et son apprentissage plusieurs questions se posent encore. En effet le calcul de l'erreur entre la sortie attendue et la sortie calculée implique la nécessité d'une fonction d'erreur. Mais aussi quelle méthode est employée pour modifier les poids ω_i et le biais b du neurone artificiel, ainsi que la valeur de la modification appliquée ? Ces questions correspondent à des hyperparamètres : la fonction d'erreur, l'optimiseur et le pas d'apprentissage qui seront détaillés dans la section 2.7.

2.2 Réseau de neurones

La section 2.1 a présenté le fonctionnement du neurone artificiel ainsi que son apprentissage. La section 2.2 détaille cette fois ci les réseaux de neurones avec le perceptron 2.2.1 et le perceptron multi couches 2.2.2. Afin de créer un réseau de neurones artificiels à partir du neurone artificiel, l'idée est de connecter les sorties d'un neurone artificiel à d'autres neurones artificiels. L'organisation de ces neurones artificiels est appelée architecture et est organisée en couche.

2.2.1 Perceptron

Le perceptron est le réseau de neurones artificiels le plus simple puisqu'il n'utilise qu'un seul neurone. La fonction d'activation du neurone est définie en fonction de l'application. Une fonction dite de tout ou rien ne sera utilisée que dans le cas de la classification alors que dans le cas de la régression, une fonction sigmoïde sera préférentiellement employée.

L'exemple ci-dessous présente l'utilisation d'un perceptron dans le but d'une classification. La figure 2.6 visualise le jeu de données utilisé. Chaque donnée est un vecteur de deux scalaires $[x_1, x_2]$ permettant de représenter un patient avec x_1 un scalaire quantifiant son niveau de pratique du sport et x_2 si le patient est un fumeur ou non. Ce jeu de données est simulé avec une fonction aléatoire centrée sur un point de l'espace. Deux classes sont alors générées sur deux points de l'espace différents. Le label de ces données est alors un risque d'avoir un cancer du poumon élevé ou faible. Le but du perceptron doit donc apprendre à prédire si un patient possède un risque élevé ou faible d'avoir un risque de cancer du poumon.

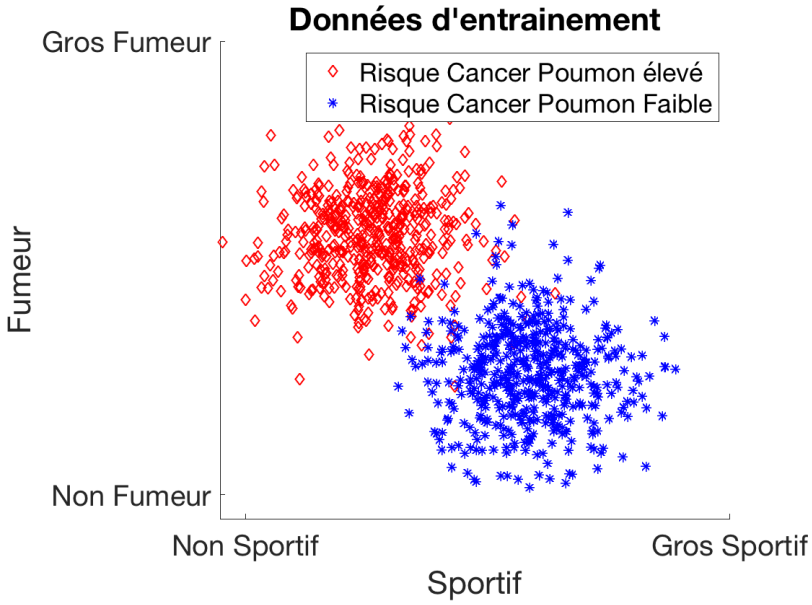


FIGURE 2.6: Jeu de données d'exemple pour l'utilisation d'un perceptron

La figure 2.7 montre l'architecture du perceptron, avec une entrée par scalaire du vecteur représentant le patient. La fonction d'activation du perceptron est une fonction dite de tout ou rien et

permet de prédire 0 ou 1.

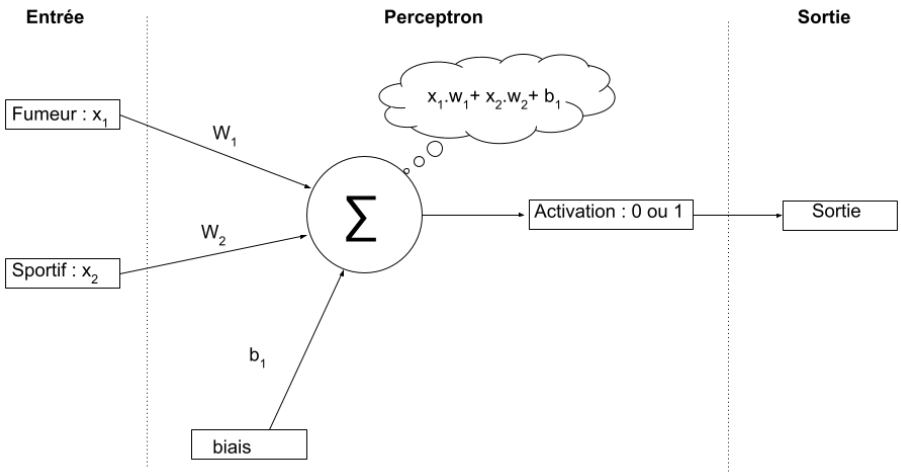


FIGURE 2.7: Architecture d'un perceptron pour une classification binaire

La simplicité du perceptron est l'un de ses avantages. Les temps de calcul sont très faibles cependant la simplicité du perceptron est aussi son inconvénient. En effet, il est possible de calculer la frontière entre les deux classes définie par un perceptron. La prédiction o correspond à l'équation 2.7.

$$\begin{cases} o = 1 \text{ si } \omega_1 \times x_1 + \omega_2 \times x_2 + b > 0 \\ o = 0 \text{ si } \omega_1 \times x_1 + \omega_2 \times x_2 + b \leq 0 \end{cases} \tag{2.7}$$

Il est alors possible de déduire la frontière entre les deux classes via l'équation 2.8.

$$\omega_1 \times x_1 + \omega_2 \times x_2 + b = 0 \tag{2.8}$$

L'équation 2.8 correspond à celle d'une droite. Le perceptron ne peut alors qu'effectuer une classification linéaire des données. Ceci explique alors la présence du biais dans les paramètres du neurone artificiel car il permet de calculer une frontière qui ne passe pas par l'origine. Afin de se convaincre de cette classification, il est possible de discrétiser le plan afin de calculer la prédiction pour chaque point du plan. La figure 2.8 montre les résultats obtenus sur les données de la figure 2.6 et permet de visualiser la classification linéaire des données.

2.2.2 Perceptron multi couches

Le perceptron multi couches est comme son nom l'indique organisé en couches. Il est alors possible de distinguer la couche d'entrée, la couche de sortie ainsi que les couches cachées. La couche d'entrée est composée d'un neurone artificiel par scalaire des données d'entrée. La couche de sortie quant à elle est composée d'un neurone par sortie. Autrement dit, dans le cas d'un problème de régression, la couche de sortie sera un unique neurone, alors que dans le cas de la classification, cette couche de sortie sera composée d'un neurone par classe.

Afin d'illustrer le perceptron multi couches, le jeu de données utilisé dans la section 2.2.1 est réutilisé avec cette fois ci trois classes. Les données sont toujours un vecteur de deux scalaires

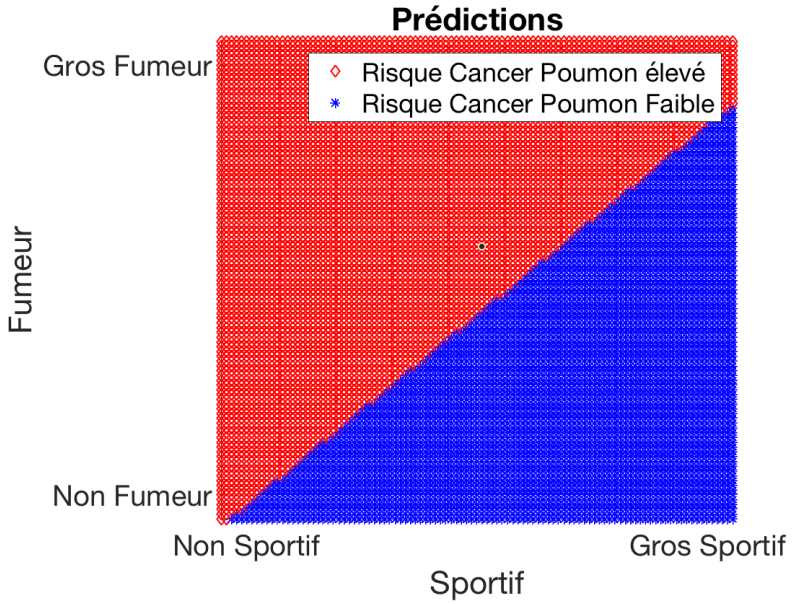


FIGURE 2.8: Prédictions calculées par un perceptron sur le jeu de données de la figure 2.6

$[x_1, x_2]$ permettant de représenter un patient avec x_1 un scalaire quantifiant son niveau de pratique du sport et x_2 si le patient est un fumeur ou non. Les classes sont maintenant un risque faible, modéré et élevé d'avoir un cancer du poumon. Ce jeu de données est présenté sur la figure 2.9.

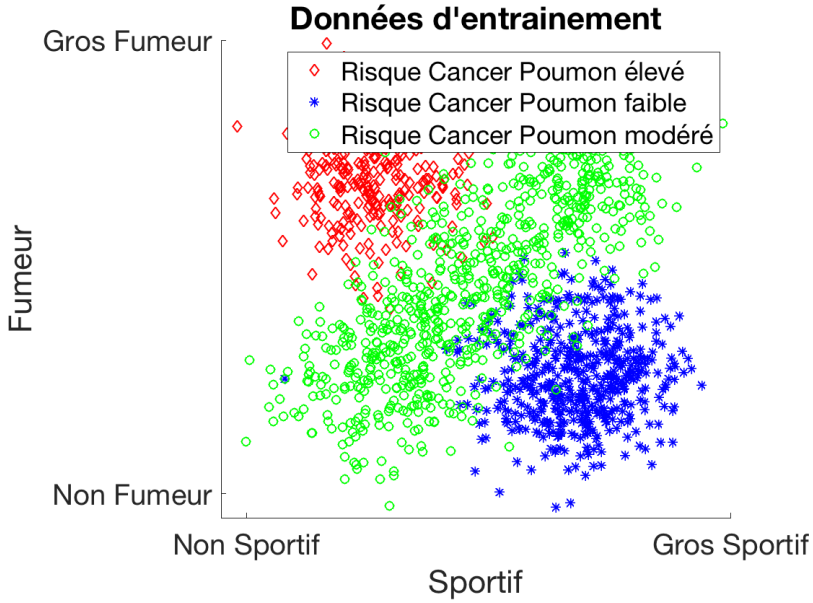


FIGURE 2.9: Jeu de données d'exemple pour l'utilisation d'un perceptron multi couches

Une architecture basée sur le perceptron multi couches est représentée sur la figure 2.10. Elle est composée d'une couche d'entrée à deux neurones artificiels, un pour x_1 et l'autre pour x_2. La couche de sortie est quant à elle composée de trois neurones artificiels, un pour la probabilité d'appartenance à la classe 1, un pour la classe 2 et un pour la classe 3. L'architecture est complétée par une couche intermédiaire, dite cachée, composée de trois neurones. Ce choix de trois neurones artificiels est un choix arbitraire, parmi les hyper-paramètres détaillés dans la section 2.7. La section 2.1.3 détaille les fonctions d'activation du neurone artificiel. Ici, la dernière couche est activée par une fonction softmax afin de calculer des probabilités d'appartenance aux classes, tandis que la couche cachée peut être activée par l'une des fonction de rectification (ReLU, LeakyReLU ou ELU).

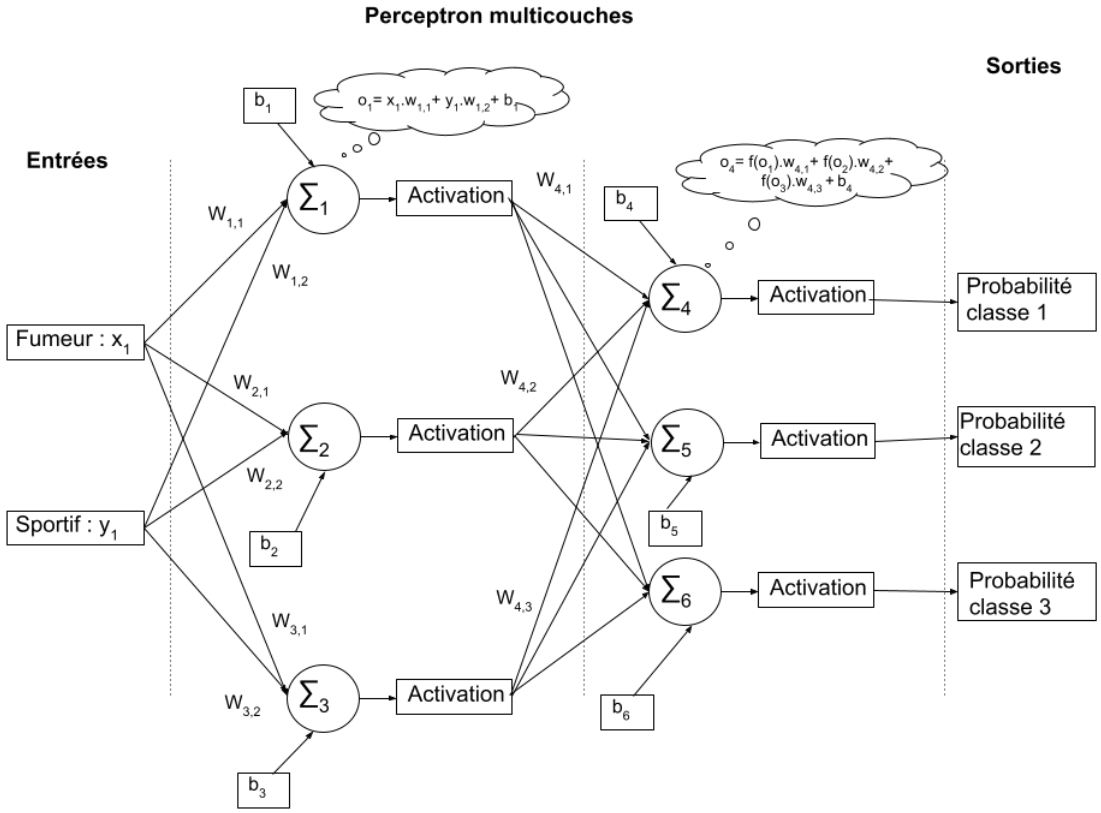


FIGURE 2.10: Architecture d'un perceptron multi couches pour une classification multi classes

Cette fois ci le perceptron multi couches permet une classification non linéaire des données. Un élément de sortie de la couche de prédiction est cette fois ci défini par l'équation 2.9 avec φ_1 la fonction d'activation de la couche cachée.

$$\varphi_1(x_1 \times \omega_{1,1} + x_2 \times \omega_{1,2} + b_1) \tag{2.9}$$

Un exemple de sortie de la couche de prédiction est cette fois ci définie par l'équation 2.10 avec

φ_2 la fonction d'activation de la couche de sortie.

$$\begin{aligned}&\varphi_2(\omega_{4,1} \times \varphi_1(x_1 \times \omega_{1,1} + x_2 \times \omega_{1,2} + b_1)\\&\quad + \omega_{4,2} \times \varphi_1(x_1 \times \omega_{2,1} + x_2 \times \omega_{2,2} + b_2)\\&\quad + \omega_{4,3} \times \varphi_1(x_1 \times \omega_{3,1} + x_2 \times \omega_{3,2} + b_3) + b_4)\end{aligned} \tag{2.10}$$

La complexité de ces équations oblige à plutôt expliquer un modèle de réseau de neurones artificiels via son architecture plutôt qu'à partir d'équations. En effet, l'équation 2.10 est déjà complexe alors que les fonctions φ_1 et φ_2 ne sont pas détaillés. De plus l'architecture est minimaliste puisqu'elle est composée de très peu de neurones artificiels. Cependant, il paraît évident que dans le cas du perceptron multi couches, la classification est cette fois ci non linéaire. La figure 2.11 présente la classification obtenue sur le jeu de données 2.9 avec l'architecture détaillée 2.10.

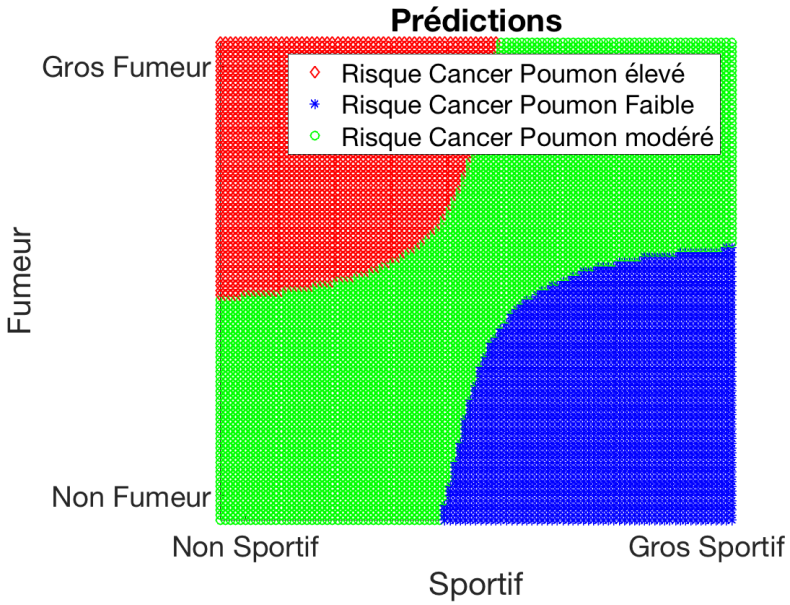


FIGURE 2.11: Prédictions calculées par un perceptron multi couches sur le jeu de données de la figure 2.9

2.3 Réseau de neurones convolutifs

Les réseaux de neurones artificiels présentés dans la section 2.2 ont l'avantage d'avoir une architecture simple. Chaque couche de neurones artificiels est entièrement connectée à ses couches adjacentes. Cependant ces architectures ne sont pas adaptées pour les données de grandes dimensions. En effet la première couche d'un perceptron multi couches est composée d'un neurone artificiel par scalaire des données d'entrée, ce qui rendrait les modèles beaucoup trop gros pour des données de plus grandes dimensions telles que les images, avec un apprentissage difficile et trop long.

Les réseaux de neurones convolutifs (CNN), toujours organisés en couches, utilisent les opérations de convolution afin de connecter les couches entre elles. En effet une couche de convolution n'est plus entièrement connectée à la suivante, mais connectée par des filtres de convolution. Ceci a pour effet

de connecter la sortie à un sous ensemble spatialement connecté des données d'entrée. L'objectif de l'apprentissage n'est alors plus de déterminer les poids de ces liens permettant de minimiser la fonction d'erreur, mais de déterminer les filtres de convolution minimisant la fonction d'erreur.

Les opérations de convolution représentée figure 2.12, permettent de détecter des formes dans un signal, ce qui permet aux réseaux de neurones convolutifs d'apprendre des formes caractéristiques sur le jeu de données d'apprentissage. Dans le but d'apprendre différentes formes, plusieurs filtres de convolution sont "appris" sur chaque couche de convolution. Il est d'usage d'augmenter le nombre de filtres de convolution au fur et à mesure que les données traversent l'architecture.

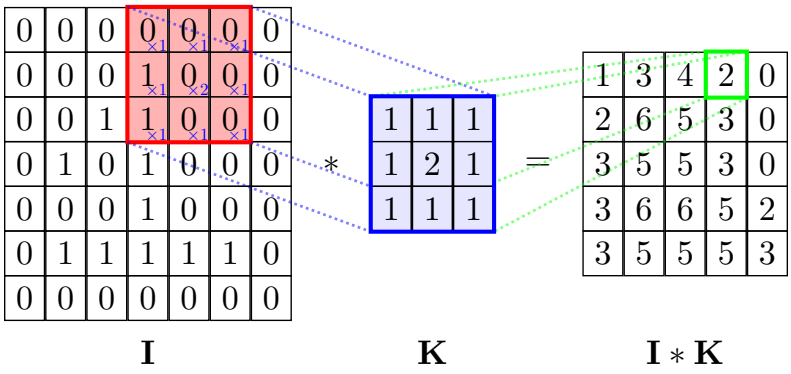


FIGURE 2.12: Détail de l'opération de convolution 2-dimensions

Les réseaux de neurones convolutifs peuvent être utilisés pour des problèmes de classification ou de régression. Dans le cas d'une classification, la fin d'un CNN est alors constituée d'un perceptron multi couches (voir section 2.2.2). Dans le cas d'un problème de régression, la sortie pourra être un perceptron multi couches si la prédiction est un scalaire, ou encore la symétrie du CNN dans le cas de segmentation (régression d'une image d'un espace $\mathbb{R}^{M \times N}$ vers un autre espace $\mathbb{R}^{M \times N}$).

Les CNN s'appuient aussi sur les opérations de "pooling" afin de réduire la dimension des données lorsqu'elles traversent le réseau de neurones. Cette opération consiste à ne garder qu'une seule valeur parmi le voisinage. Cette valeur peut être le maximum ou la moyenne du voisinage. La figure 2.13 présente un exemple de pooling utilisant la moyenne.

Finalement, les CNN sont principalement basés sur une séquence de couches de convolution avec un nombre de filtres appris croissant, et de couches de pooling pour réduire la dimension des données. Les sections 2.3.1, 2.3.2 et 2.3.3 présentent les architectures les plus connues et utilisées dans la littérature. La section 2.3.4 quant à elle présente l'architecture la plus connue dans un but de segmentation d'images. Ces architectures sont disponibles déjà entrainées sur le jeu de données Image-net [17], l'un des plus gros jeu de données d'images couleur contenant de nombreuses classes. Les filtres de convolution et les paramètres de ces CNN sont alors déjà fixés pour minimiser une fonction d'erreur sur un grand nombre de classes. Ces CNN sont donc capables de détecter de nombreuses formes sur les images et peuvent être réutilisés pour d'autres applications.

Les 3 principales architectures actuellement utilisées dans la littérature sont détaillé ci-dessous.

2.3.1 VGG

L'architecture VGG pour Visual Geometry Group a été développée par l'Université d' Oxford et est l'une des premières architectures CNN présentées dans la littérature [18]. Son architecture

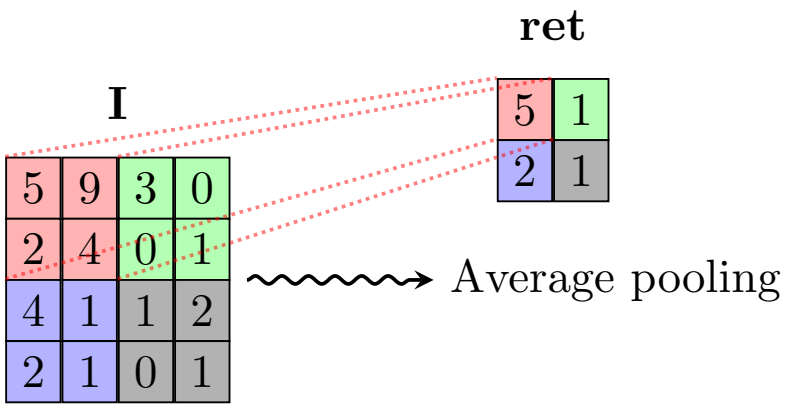


FIGURE 2.13: Détail de l'opération d'average pooling

est une séquence de couches de convolution et de pooling. La figure 2.14 détaille l'architecture d'un modèle VGG. Il existe deux versions VGG16 et VGG19 qui correspondent aux nombres de couches utilisées.

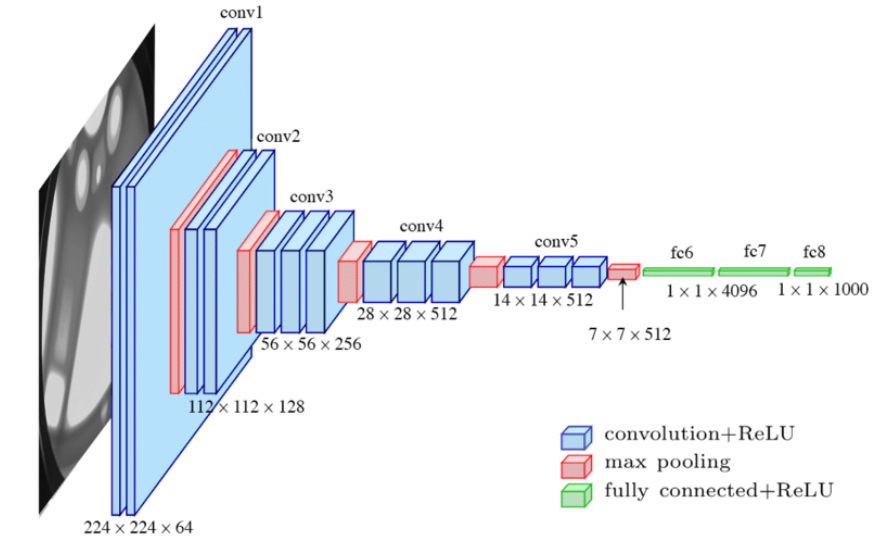


FIGURE 2.14: Architecture du modèle VGG 16

L'architecture du modèle présenté figure 2.14 est donc :
1 Convolution 64 filtres, kernel $(3, 3)$ + activation ReLU
2 Convolution 64 filtres, kernel $(3, 3)$ + activation ReLU
3 Max Pooling
4 Convolution 128 filtres, kernel $(3, 3)$ + activation ReLU
5 Convolution 128 filtres, kernel $(3, 3)$ + activation ReLU

6 Max Pooling
7 Convolution 256 filtres, kernel $(3, 3)$ + activation ReLU
8 Convolution 256 filtres, kernel $(3, 3)$ + activation ReLU
9 Convolution 256 filtres, kernel $(3, 3)$ + activation ReLU
10 Max Pooling
11 Convolution 512 filtres, kernel $(3, 3)$ + activation ReLU
12 Convolution 512 filtres, kernel $(3, 3)$ + activation ReLU
13 Convolution 512 filtres, kernel $(3, 3)$ + activation ReLU
14 Max Pooling
15 Convolution 512 filtres, kernel $(3, 3)$ + activation ReLU
16 Convolution 512 filtres, kernel $(3, 3)$ + activation ReLU
17 Convolution 512 filtres, kernel $(3, 3)$ + activation ReLU
18 Max Pooling
19 Mise à plat
20 Couche entièrement connectée de 4096 neurones + activation ReLU
21 Couche entièrement connectée de 4096 neurones + activation ReLU
22 Couche entièrement connectée de 1000 neurones + activation softmax

L'avantage des modèles VGG est leur "simplicité". Cependant, cette simplicité dans la conception de l'architecture a des conséquences telles que la quantité de mémoire utilisée pour ces types de modèle.

2.3.2 ResNet

Les architectures basées sur ResNet ressemblent aux modèles VGG mais un lien permet de sauter certaines couches via les "skip connexions" [19]. La figure 2.15 présente le principe de la skip connection. Un lien avec une couche identité est reconnectée plus profondément dans l'architecture ce qui permet aux données de "sauter" une couche lorsqu'elle est inutile dans les prédictions.

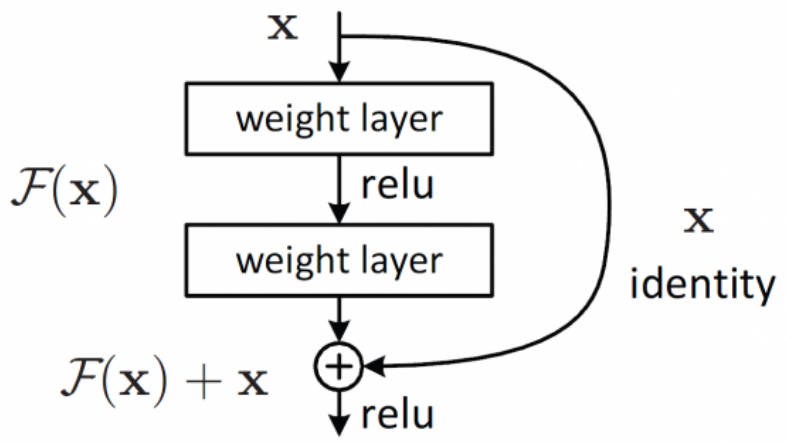


FIGURE 2.15: Skip connexion du modèle ResNet

La figure 2.16 représente la skip connexion de façon simplifiée où la couche $i - 2$ est à la fois connectée à la couche $i - 1$, comme ce serait le cas pour une architecture VGG, mais aussi à la couche i.

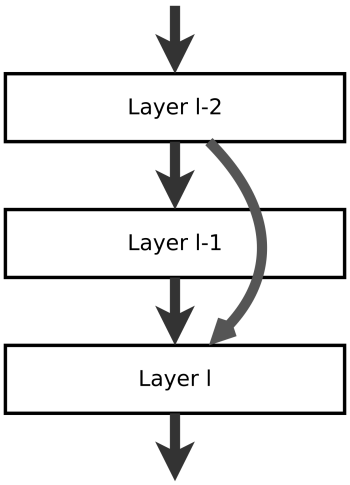


FIGURE 2.16: Architecture des couches de convolution des modèles type ResNet

L'apprentissage des modèles ResNet est alors facilité car le modèle peut choisir les couches qui lui sont utiles pour minimiser l'erreur de prédiction. Différentes versions des architectures ResNet sont disponibles dans la littérature avec des paramètres déjà entrainés sur le jeu de données Image-net [17], allant de ResNet50 à ResNet151.

2.3.3 GoogLeNet

Les architectures de type GoogLeNet utilisent les "modules d'inception" pour remplacer les simples couches de convolution [20]. La figure 2.17 détaille le module d'inception et il est possible de remarquer qu'il ne s'agit plus d'une séquence linéaire de couches. En effet le module d'inception combine les convolutions de taille $(1,1)$, $(3,3)$ et $(5,5)$ ainsi que l'opération de max pooling. Les convolutions de taille $(1,1)$ permettent de pondérer les données de la couche précédente, tandis que les couches de convolution $(3,3)$ et $(5,5)$ permettent d'apprendre des formes sur les données en prenant en compte un voisinage plus ou moins grand. L'opération de pooling inclue dans le module d'inception permet quant à elle d'avoir un aperçu de ce qui se passera lors de la couche suivante. Le module d'inception se termine par une concaténation de chacune des branches afin de pouvoir être connecté à la couche suivante. Finalement le module d'inception combine une pondération des données, un aperçu de la couche suivante ainsi que deux convolutions de taille différente.

La figure 2.18 détaille l'architecture du modèle GoogLeNet utilisant les modules d'inception. Il est possible de constater que l'architecture présente plusieurs sorties afin d'obtenir une prédiction à partir de formes extraites sur les données d'apprentissage provenant de différentes échelles.

2.3.4 U-Net

Les architectures basées sur les U-Net sont utilisées dans le cas de régression ou de segmentation. Comme son nom l'indique, l'architecture est en forme de U. Une première partie consiste en des couches de convolutions et de pooling, puis la seconde partie est la symétrie de la première. Des couches de déconvolution avec des nombres de filtres égaux à la couche équivalente de la première partie et des couches d'"un-pooling" permettent d'augmenter la dimension des données transitant à

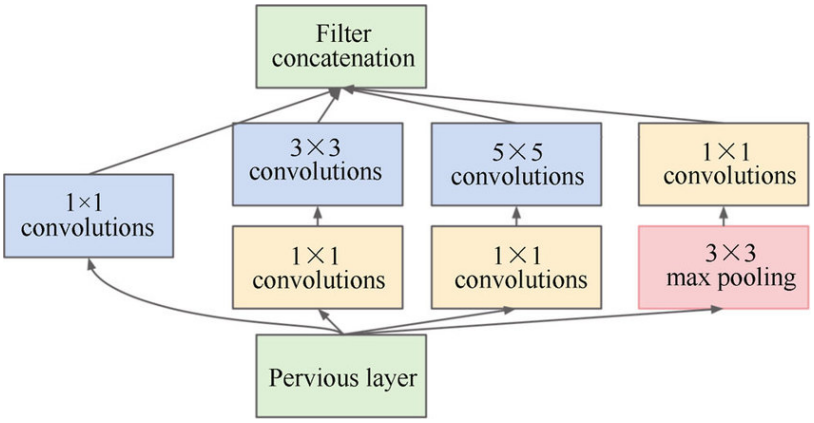


FIGURE 2.17: Module d'inception du modèle GoogLeNet

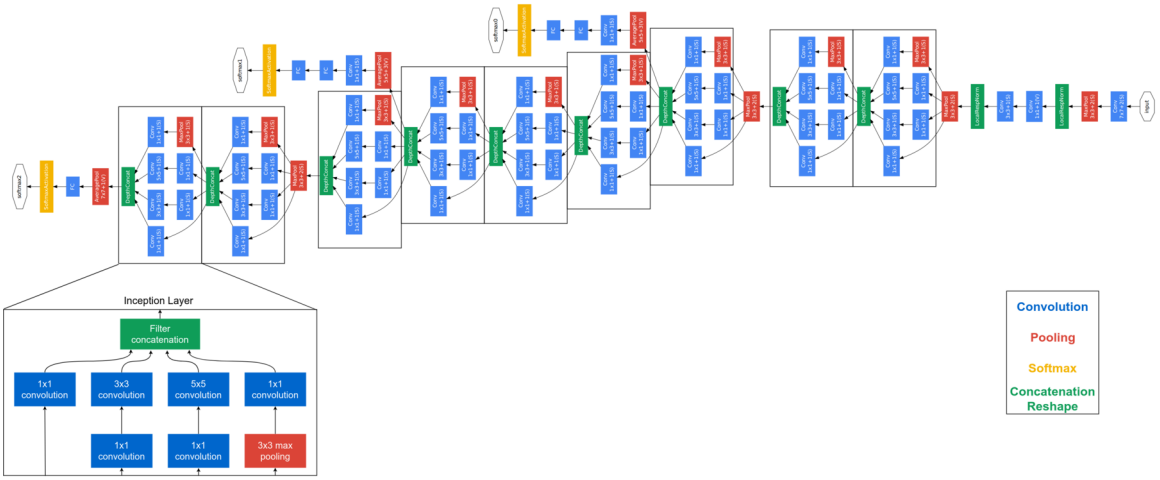


FIGURE 2.18: Architecture du modèle GoogLeNet

l'intérieur du U-Net. Les deux parties peuvent être connectées avec ou sans l'utilisation de couches entièrement connectées. La figure 2.19 détaille de l'architecture d'un U-Net.

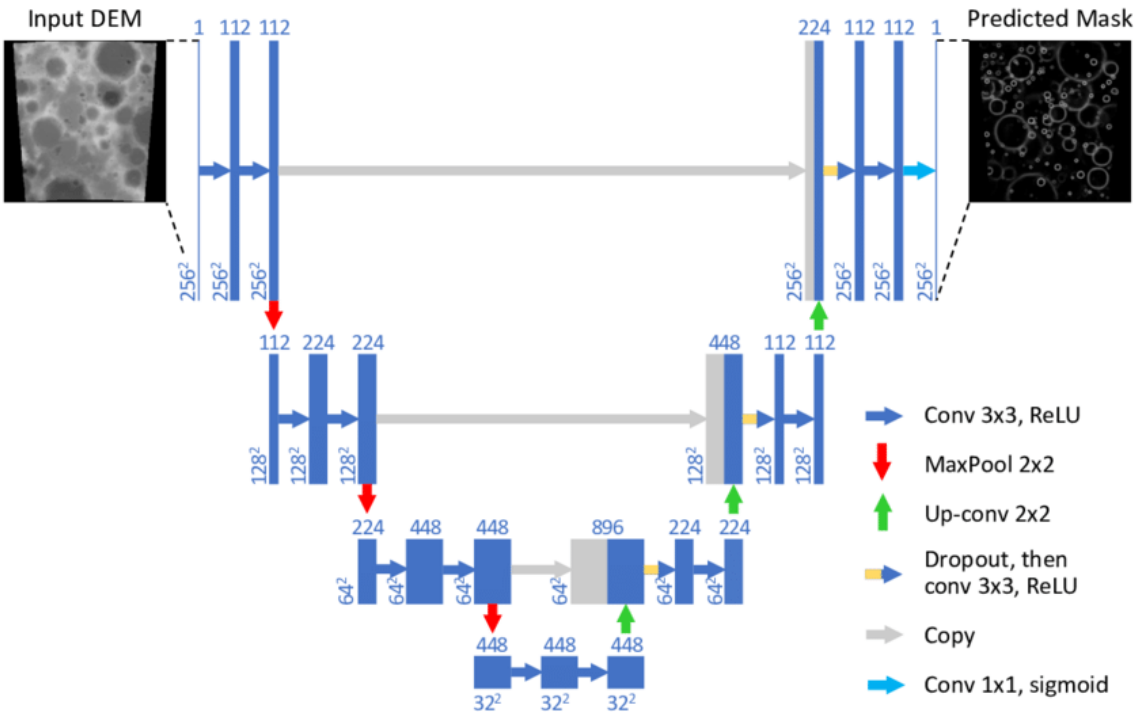


FIGURE 2.19: Architecture du modèle U-Net

2.4 Transfer learning

Le transfer learning consiste à utiliser un modèle déjà entrainé sur un jeu de données pour une application précise, mais cette fois-ci sur un autre jeu de données ou une autre application [67].

Les modèles VGG [18], ResNet [19] et GoogLeNet [20] présentés dans la section 2.3 ont tous été entrainés sur le jeu de données Image-net [17]. Ces modèles sont alors disponibles avec les paramètres déjà entrainés. Le jeu de données Image-net étant composé d'un grand nombre d'images et de classes, ces modèles ont alors eu l'occasion d'apprendre à reconnaitre de nombreuses formes et caractéristiques dans une image. Ils peuvent alors être appliqués à d'autre domaine tant que les données d'entrée sont des images.

En général, la partie convolutive de ces modèles va être fixée, tandis que les couches entièrement connectées vont être ré-entrainées sur le nouveau jeu de données. Pour plus d'information sur le transfer learning voir [68].

2.5 Data augmentation

La quantité de données est un facteur important dans la réussite du développement d'un modèle de deep learning. Une technique couramment utilisée dans le deep learning pour augmenter la taille du jeu de données est la data augmentation qui vise à créer plusieurs exemples différents d'une seule

et même donnée.
Plusieurs méthodes sont alors à disposition telles que l'ajout de bruit aléatoire, les rotations, les translations ou encore les miroirs. Cependant ces méthodes ne sont pas toujours applicables et nécessitent une étape de réflexion sur la modification applicable qui permet de conserver le sens du jeu de données.
En prenant l'exemple d'images satellites visant à détecter les routes, la plupart des méthodes sont applicables. En effet, une route dans une image reste une route lorsqu'une translation, une rotation ou un image miroir est générée. Cependant, dans le cas de reconnaissance des chiffres manuscrits, une rotation de 180 degrès d'une image du chiffre "6" ne produit pas un nouvel exemple d'image de cette classe mais une image du chiffre "9" qui serait alors labélisée comme étant un chiffre 6. Il est bien possible d'appliquer des rotations sur un jeu de données de chiffres manuscrits puisqu'il est possible de rédiger avec une inclinaison différentes tout en gardant le sens, mais il existe un angle de rotation à ne pas dépasser au risque de fausser la base de données d'apprentissage.
Concernant le cas la méthode de l'ajout de bruit, elle est toujours applicable mais elle nécessite d'analyser le type de bruit qu'il est possible d'ajouter ainsi que le nombre de nouvelles données à générer. En effet, tous les capteurs possèdent un bruit lors de l'acquisition, il est donc possible de générer plusieurs images simulant plusieurs acquisitions d'une même donnée avec différentes manifestations du bruit du capteur.
Dans le cas des images médicales, et plus particulièrement des images neurologiques, l'application des méthodes de data augmentation est difficile. En effet, le jeu de données étant recalé dans un template MNI (voir la section 1.2.3), les techniques de rotations, de miroir et de translation ne sont pas indiquées. Cela est d'autant plus contre-indiqué lorsqu'une signature spatiale est recherchée. L'ajout de méthode de data augmentation pourrait fausser les résultats de visualisation. En effet, lorsque l'on applique une image miroir d'une IRM cérébrale, la droite et la gauche sont alors confondues et un modèle d'intelligence artificielle n'est plus capable de discerner la droite et la gauche dans une IRM cérébrale.

2.6 Mesure des performances d'un modèle

Afin d'estimer la performance d'un modèle, plusieurs solutions sont disponibles selon le type d'application de ce dernier. Dans le cas d'une régression, la fonction d'erreur sera prise en compte tandis que dans le cas d'une classification il est possible d'utiliser en plus l'exactitude (ou accuracy). Afin de pouvoir évaluer les performances d'un modèle sur des données qu'il n'a jamais vues, il est d'usage de garder une partie de celle-ci pour le test. Il est aussi habituel de conserver une partie des données afin de mesurer les performances du modèle sur des données qu'il n'utilise pas pour apprendre, mais simplement pour vérifier son comportement sur des données inconnues. De telles données sont appelées données de validation et permettent de réctifier certains hyperparamètres d'apprentissage. Il est courant de voir une découpe aléatoire du jeu de données en 70% pour l'entrainement, 15% pour la validation et 15% pour le test.

2.6.1 Perte

La perte, ou erreur, d'un modèle vise à calculer l'erreur du modèle sur l'ensemble des données testées. Pour cela il est nécessaire de définir une fonction d'erreur permettant de quantifier l'erreur entre la prédiction et la vérité terrain. Un exemple d'une telle fonction dans le but d'une classification est l'entropie croisée dont l'équation 2.11 détaille le calcul, avec K le nombre de classes, y_i la probabilité d'appartenance à la classe i de la vérité terrain (i.e. 0 ou 1) et $\hat{y_i}$ la probabilité

d'appartenance à la classe i prédite par le modèle. D'autres fonctions d'erreurs seront détaillées dans la section 2.7.6.

$$logloss(y, \hat{y}) = \sum_{i=0}^{i=K} -(y_i \times log(\hat{y_i}) + (1 - y_i) \times log(1 - \hat{y_i})) \tag{2.11}$$

Cette fonction d'erreur est minimisée lors de l'apprentissage sur les données d'entrainement dans le but d'être par la même occasion minimisée sur les données de test qui n'ont jamais été vues par le modèle durant la phase d'entrainement.

La figure suivante 2.20 présente une fonction d'erreur sur les données d'apprentissage et de test lors de l'entrainement d'un modèle. Il est alors possible de constater que la courbe décroit de façon exponentielle. De plus, la comparaison entre la courbe correspondant aux données d'entrainement et celle des données de test permet de repérer les cas de sur-apprentissage et de sous-apprentissage.

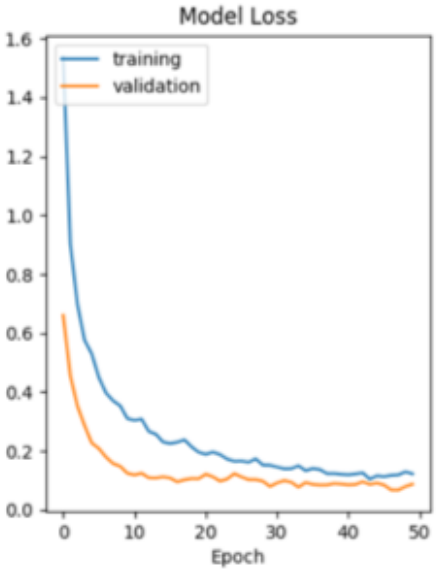
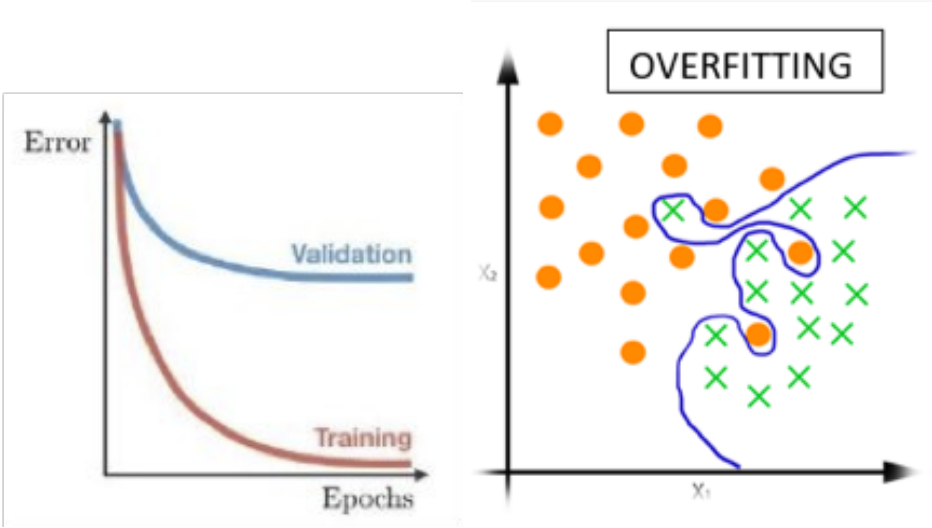


FIGURE 2.20: Courbe d'apprentissage de la fonction d'erreur sur le jeu de données d'entrainement et de test

Le sur-apprentissage correspond au cas où la courbe d'erreur est très faible sur les données d'entrainement mais reste élevée sur les données de test. Cela correspond au fait que le modèle est devenu trop "performant" sur les données d'apprentissage et ne généralise plus correctement l'information contenue dans l'ensemble du jeu de données, mais au contraire s'appuie sur des caractéristiques présentes uniquement dans les données d'entrainement. La figure 2.21 permet de visualiser le cas du sur-apprentissage sur la courbe de la fonction d'erreur et son effet sur la classification du jeu de données.

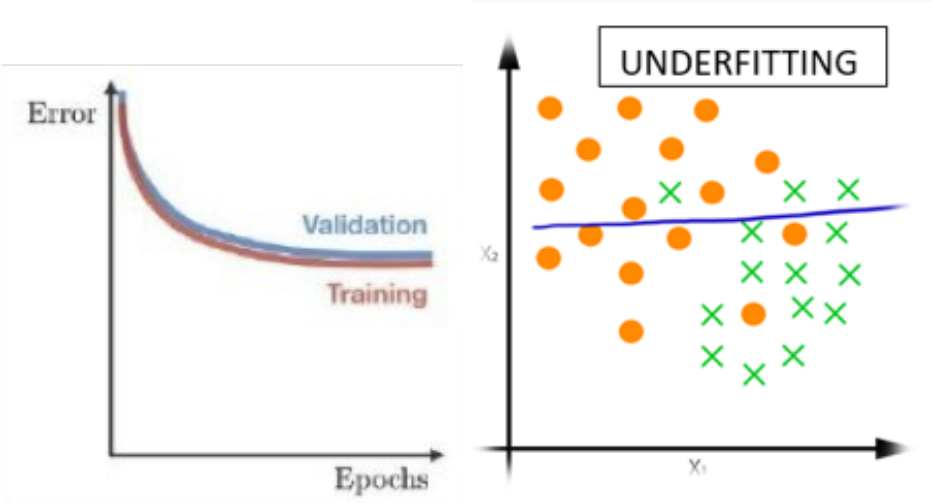
A contrario, le sous apprentissage est un modèle qui obtient deux courbes d'erreur sur les données d'entrainement et de test qui reste élevée tout au long de l'apprentissage. La figure 2.22 permet de visualiser le cas du sous-apprentissage sur la courbe de la fonction d'erreur et son effet sur la classification du jeu de données.



(a) Cas de sur-apprentissage sur la courbe d'apprentissage de la fonction d'erreur

(b) Représentation d'un cas de sur-apprentissage sur les données

FIGURE 2.21: Détection et représentation du sur-apprentissage



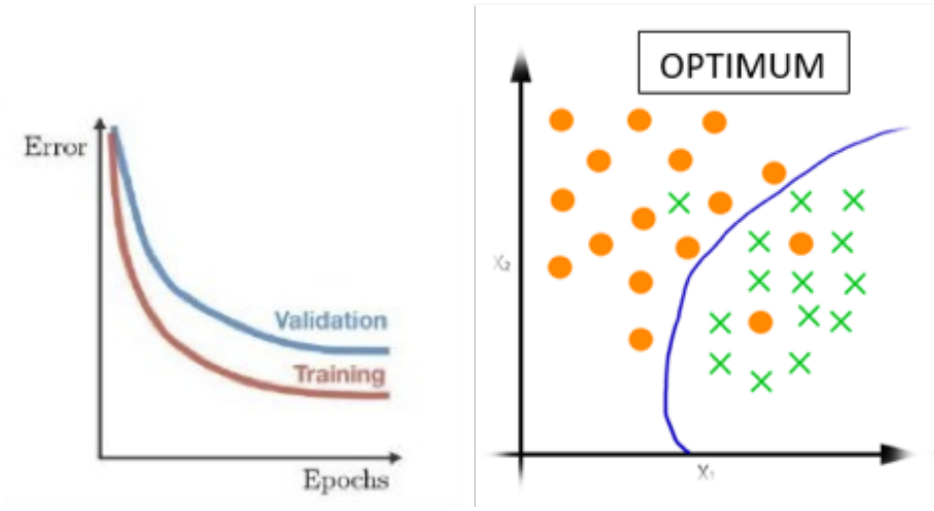
(a) Cas de sous-apprentissage sur la courbe d'apprentissage de la fonction d'erreur

(b) Représentation d'un cas de sous-apprentissage sur les données

FIGURE 2.22: Détection et représentation du sous-apprentissage

La figure 2.22 permet de visualiser le cas du sous-apprentissage sur la courbe de la fonction d'erreur et son effet sur la classification du jeu de données.

Plusieurs techniques sont disponibles afin de converger vers le cas idéal. Il est possible de citer l'utilisation de couches de "dropout" (détaillé dans la section 2.6.3) ou de régulariseur pour éviter



(a) Cas idéal d'apprentissage sur la courbe d'apprentissage de la fonction d'erreur

(b) Représentation d'un cas idéal d'apprentissage sur les données

FIGURE 2.23: Détection et représentation du cas idéal d'apprentissage

le sur-apprentissage, et complexifier le modèle afin d'éviter le sous-apprentissage.

2.6.2 Exactitude

Dans le cas d'un problème de classification, une autre mesure de performance est à disposition. L'exactitude (ou accuracy) correspond au pourcentage de données correctement classées par le modèle. En définissant les vrais positifs et les vrais négatifs (VP et VN) comme les données correctement classées par le modèle et les faux positifs et les faux négatifs (FP et FN) comme les données mal classées par le modèle, il est possible de calculer un score d'exactitude. Dans le cas d'une classification binaire appliquée au domaine médical telle que sujets sains et patients malades, un VP est un patient malade prédit comme tel par le modèle et un VN est un sujet sain prédit comme sujet sain. Un FP est un sujet sain prédit comme patient malade et un FN est un patient malade prédit comme sujet sain. L'équation 2.12 détaille le calcul permettant d'obtenir le score d'exactitude sur un jeu de données.

$$\text{Exatitude} = \frac{VP + VN}{VP + VN + FP + FN} \tag{2.12}$$

Ainsi il est possible de calculer ce score d'exactitude lors de l'apprentissage et la figure 2.24 en montre un exemple. Contrairement à la fonction de perte, la courbe d'accuracy augmente de façon logarithmique.

Tout comme pour la fonction de perte (section 2.6.1), il est possible de repérer le sur-apprentissage et le sous-apprentissage sur ces courbes en comparant l'exactitude obtenue sur les données d'entrainement et celle des données de test (voir figures 2.22, 2.21 et 2.23)

Cependant, ce score d'exactitude présente des limites puisqu'il correspond à un score général. Pour rester dans l'exemple de la classification binaire appliquée au domaine médical, ce score ne reflète pas la capacité du modèle à classer correctement les sujets sains ou au contraire, les patients malades. Pour cela, il est possible d'étudier les scores de sensitivité et de spécificité correspondant à

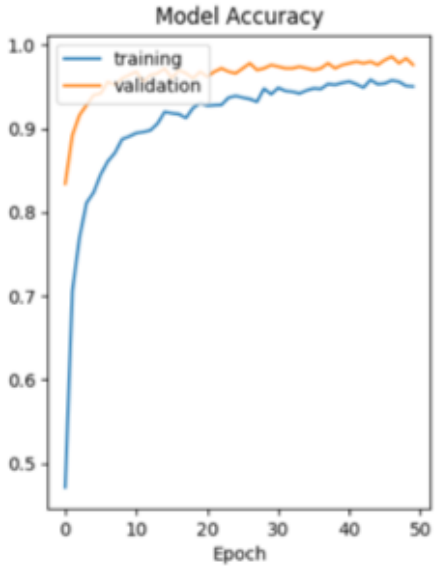


FIGURE 2.24: Courbe d'apprentissage de l'exactitude sur le jeu de données d'entrainement et de test

la capacité du modèle à classer correctement respectivement les patients malades et les sujets sains. Ces deux scores permettent d'affiner le modèle en fonction des besoins de l'utilisateur. Les équations 2.13 et 2.14 détaillent les calculs utilisés afin d'obtenir ces deux scores.

$$\text{Sensitivité} = \frac{VP}{VP + FN} \tag{2.13}$$

$$\text{Spécificité} = \frac{VN}{VN + FP} \tag{2.14}$$

2.6.3 Amélioration des performances

Hormis le choix des hyper-paramètres du modèle définis en section 2.7, plusieurs techniques sont disponibles afin d'améliorer les performances d'un modèle. En effet pour éviter le sous-apprentissage, un modèle peut être complexifié afin de modéliser de façon plus complexe le problème.
Dans le cas du sur-apprentissage, il est possible d'appliquer du dropout ce qui a pour effet de désactiver certains neurones lors de la phase d'apprentissage [69]. L'apprentissage est alors plus long, mais il est aussi plus robuste et moins sensible au bruit présent dans les données d'apprentissage. La figure 2.25 présente un exemple de dropout sur une couche entièrement connectée. L'utilisateur définit une probabilité de désactivation d'un neurone pour une couche, et un tirage aléatoire permet de désactiver les neurones ayant une probabilité p inférieure à la valeur définie par l'utilisateur.

Une autre méthode de prévention du sur-apprentissage est l'application d'un régulariseur qui a pour effet de lisser les valeurs des poids et filtres de convolution en ajoutant à l'erreur la norme des poids. La fonction d'erreur entre y et $\hat{y}$ est alors définie par l'équation 2.15, avec ω_i les poids ou

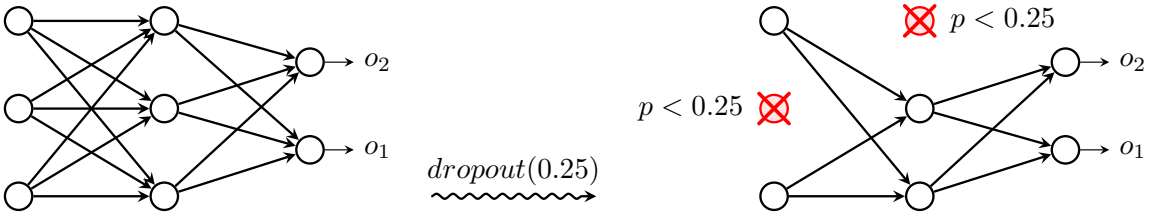


FIGURE 2.25: Exemple de l'application du dropout sur une couche entièrement connectée

filtres de convolution de la couche sur laquelle est appliquée le régulariseur.

$$loss(y, \hat{y}) = loss(y, \hat{y}) + \sum_{i=0}^{N} \|\omega_i\| \tag{2.15}$$

2.7 Hyper-paramètres d'un réseau de neurones

Après avoir détaillé le neurone artificiel en section 2.1, mais aussi les réseaux de neurones entièrement connectés et convolutifs en sections 2.1 et 2.3 ainsi que les mesures de performance d'un modèle en section 2.6, la section 2.7 présente les différents hyper-paramètres d'un modèle. Ces hyper-paramètres sont à définir par l'utilisateur afin d'obtenir un modèle compatible et performant pour l'application souhaitée.

Pour cela, l'utilisateur va devoir choisir un type d'architecture puis affiner ces hyper-paramètres en s'appuyant sur les mesures de performance, mais aussi les dimensions des données d'entrée ainsi que la taille du modèle et la capacité de la machine de calcul.

2.7.1 Nombre de couches entièrement connectées

L'un des paramètres à définir une fois que l'architecture est choisie, est le nombre de couches qui seront utilisées. Pour cela, l'utilisateur va s'appuyer à la fois sur la dimension des données d'entrée mais aussi sur les mesures de performance du modèle.

En utilisant plus de couches, la solution vers laquelle va converger le modèle lors de l'apprentissage va être plus complexe et moins linéaire. L'utilisateur pourrait choisir un grand nombre de couches, cependant le temps de calcul est augmenté et un risque de sur-apprentissage est présent. De plus, un trop grand nombre de couches augmenterait le risque de dépasser la capacité de la machine de calcul.

Au contraire, en choisissant un nombre faible de couches, la solution proposée par le modèle va être plus simple et donc nécessiter moins de temps de calcul et une machine de calcul moins puissante. Cependant un modèle trop simple risque de converger vers un sous-apprentissage.

Finalement, l'utilisateur doit trouver le bon équilibre permettant d'obtenir de bons résultats sur les mesures de performance du modèle.

2.7.2 Nombre de neurones

Le nombre de neurones utilisés pour les couches entièrement connectées est un autre hyper-paramètre à définir par l'utilisateur. En combinant plusieurs couches et un nombre élevé de neurones par couche, une nouvelle fois la solution proposée par le modèle va être plus complexe, mais sera plus

longue à trouver. De plus le nombre de neurones sur une couche entièrement connectée conduit a un modèle très gourmand en mémoire et est l'une des principales sources de dépassement de la capacité d'une machine de calcul. En effet en prenant l'exemple de deux couches entièrement connectées de 1000 neurones chacune, chacun des 1000 neurones de la première couche possède 1000 connections vers les 1000 neurones de la couche suivante. Chacune de ces connections correspond à un poids de type $\mathbb{R}$ codé sur 16 ou 32 bits selon les réglages effectués par l'utilisateur. Ce qui revient à $1000 \times 1000 = 1000000$ de poids à stocker, mais aussi à déterminer lors de l'apprentissage.

Aussi l'utilisateur doit veiller à ne pas faire de goulot d'étranglement trop sévère, sauf cas particulier, en évitant de connecter une couche entièrement connectée avec un grand nombre de neurones sur une autre couche possédant quant à elle peu de neurones.

Finalement, en utilisant ces indices sur la capacité de la machine de calcul, la dimension des données d'entrée ainsi que le nombre de sorties du modèle, l'utilisateur peut converger vers une architecture permettant d'obtenir un bon compromis entre les résultats, la complexité du modèle et la capacité de la machine de calcul.

2.7.3 Nombre de couches de convolution

Afin de déterminer un nombre de couches de convolution convenant au problème, l'utilisateur peut s'appuyer sur la dimension des données d'entrée. En effet, les couches de convolution étant séquentielles, la sortie d'une couche de convolution se retrouve elle-même convoluée. Ainsi, au fur et à mesure de l'enchainement des couches de convolution, un pixel agrège de l'information provenant d'un voisinage de plus en plus grand. La figure 2.26 illustre ce phénomène avec la donnée 2-dimensions I_0 convoluée trois fois de suite. Le pixel bleu de la donnée I_3 contient de l'information en relation avec la totalité de la donnée I_0. L'ajout de couches de convolution au-delà aura pour effet d'ajouter de l'information provenant du "padding" de la convolution, qui consiste à augmenter la dimension d'une donnée afin de pouvoir appliquer une opération de convolution, en réduisant ainsi la proportion d'information provenant de la donnée d'entrée.

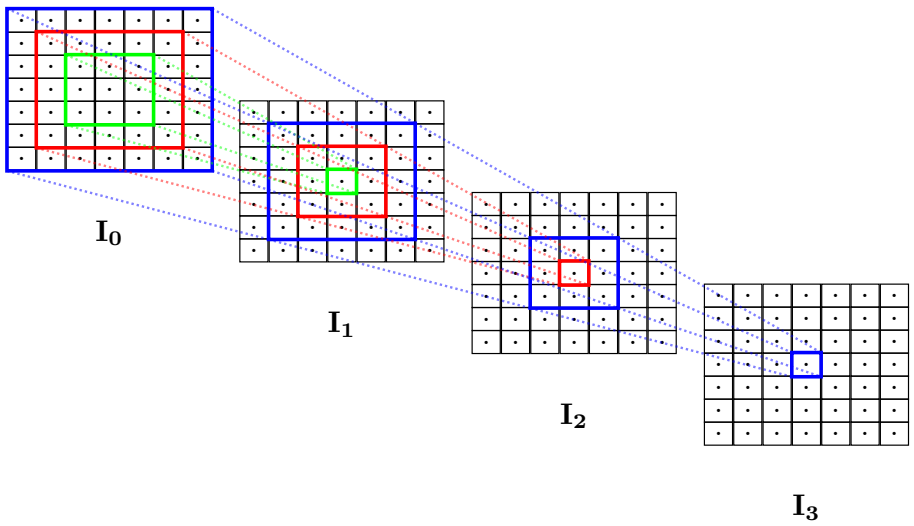


FIGURE 2.26: Enchainement de plusieurs convolutions

Ce phénomène est d'autant plus amplifié que le modèle inclut des couches de pooling qui ré-

duisent la dimension des données.

L'utilisation de la convolution "valide" n'applique pas de padding et a pour effet de réduire les données en fonction de la taille du filtre (la dimension d'une donnée convoluée par un filtre de taille $N+1$ est réduite de N), permet aussi de déterminer un nombre de couches de convolution convenable. En effet, lorsqu'il n'y a pas de padding, une opération de convolution ne peut pas s'appliquer sur des données de dimension strictement inférieure à la taille du filtre. Ainsi à partir d'un certain nombre de couches de convolution avec l'option "valide", il n'est plus possible d'en ajouter sous peine d'avoir un modèle incapable d'être compilé.

L'utilisateur peut aussi s'appuyer sur la physique associée au problème afin de déterminer le nombre idéal de couches. En prenant l'exemple d'un modèle ayant pour but de modéliser une diffusion, en disposant de la taille physique du phénomène de diffusion ainsi que de la taille des pixels, il est possible de déduire un nombre de couches de convolution permettant d'agréger l'information provenant d'un voisinage correspondant à la physique du phénomène de diffusion à modéliser.

Finalement l'utilisateur peut s'appuyer sur le fonctionnement de l'opération de convolution ainsi que sur les informations a priori du problème pour converger vers un nombre de couches de convolution adapté au problème.

2.7.4 Taille et nombre des filtres de convolution

La taille et le nombre de filtres de convolution par couche est un autres hyper-paramètres à déterminer. La taille du filtre de convolution augmente la taille du voisinage prise en compte lors de la convolution, cependant les modèles étant composés d'une séquence de couches de convolution, le voisinage pris en compte augmente au fur et à mesure de l'enchainement des couches de convolution. Il n'est alors pas nécessaire d'avoir des filtres de convolutions de trop grande taille. Dans la littérature il est souvent utilisé des filtres de taille 3 ou 5 voir 7 pour la plupart des architectures. Le fait d'augmenter la taille des filtres de convolution implique une nette augmentation de la taille du modèle, ainsi que sa complexité ce qui a pour effet de détériorer l'apprentissage. Une taille de filtres de convolution impaire permet de centrer le filtre sur le pixel courant, c'est pourquoi les filtres de taille paire ne sont que très rarement utilisés.

Il est d'usage d'augmenter le nombre de filtres de convolutions au fur et à mesure que les couches de convolution s'enchainent. L'utilisateur doit alors principalement déterminer le nombre de filtres de convolution de la première couche. Or la première couche de convolution fait intervenir les données lorsqu'elles sont dans leur plus grande dimension. Les capacités de la machine de calcul sont alors un bon moyen de déterminer le nombre de filtres de convolution de la première couche. La plupart des modèles présents dans la littérature tels que les modèles VGG [18], ResNet [19] et GoogLeNet [20] utilisent 64 filtres pour la première couche de convolution.
Pour les modèles utilisant des couches entièrement connectées suite à la partie convolutive, il est possible de s'appuyer une nouvelle fois sur les capacités de la machine de calcul pour choisir le nombre de filtres de la dernière couche. En effet lorsque la sortie de la dernière couche de convolution est mise à plat, les données de dimension (x, y, f) vont être mises à plat sous forme d'un vecteur de dimension $x \times y \times f$ avec f le nombre de filtres de la dernière couche de convolution. Il est alors nécessaire que la valeur de $x \times y \times f$ ne soit pas trop élevée pour faciliter l'apprentissage sur les couches entièrement connectées. Pour cela, il faut que f ne soit pas trop élevé ; 256 ou 512 sont couramment utilisés dans la plupart des modèles de la littérature. Une autre possibilité est d'avoir

suffisamment réduit la dimension (x, y) des données via le nombre de couches de convolution valides ou les opérations de pooling.

2.7.5 Initialisation

L'initialisation des paramètres du réseau de neurones est une étape importante du développement d'un modèle. En effet, plusieurs méthodes sont disponibles et ont chacune leurs avantages. L'intérêt d'évaluer l'effet de l'initialisation des paramètres sur les résultats prédits par un modèle permet de confirmer que la solution proposée par un modèle est indépendante de son initialisation. Autrement dit, qu'un modèle converge toujours vers des solutions similaires quelle que soit l'initialisation.

Deux initialisations "naïves" sont possibles. Initialiser tous les paramètres du modèle à 0 ou à 1 (ou éventuellement toute autre valeur). Cependant, en utilisant ce type d'initialisation il est difficile d'écarter le cas où la solution trouvée par un modèle n'est pas déterminée par l'initialisation. De plus, l'apprentissage d'un modèle utilisant ce type d'initialisation n'est pas favorisé.

Deux initialisations plus élaborées sont plus communément utilisées en intelligence artificielle. Une initialisation suivant une loi normale dont les paramètres du centre et de la déviation standard peuvent être définis par l'utilisateur permet de commencer l'apprentissage d'un modèle à partir d'une solution aléatoire. Chaque poids ω est alors initialisé suite à un tirage aléatoire $\omega \sim \mathcal{N}(\mu, \sigma^2)$ avec la moyenne μ et la variance σ^2 pouvant être modifiées par l'utilisateur.
Sur le même principe, il est possible d'utiliser une initialisation uniforme où l'utilisateur peut préciser les bornes de la loi uniforme. Chaque poids ω est alors initialisé entre $[min \dots max]$ ou chaque valeur présente dans l'intervalle est équiprobable, avec min et max pouvant être définis par l'utilisateur.

D'autres initialiseurs sont aussi disponibles dans la littérature et ont l'avantage de calculer les paramètres des lois d'initialisation automatiquement de façon à faciliter l'apprentissage du modèle. Le plus utilisé est alors l'initialiseur Glorot [70] et est disponible en version loi normale et loi uniforme. Dans le cas de la loi normale, chaque poids ω est initialisé en suivant une loi $\omega \sim \mathcal{N}(0, \sigma^2)$ avec :

$$\sigma^2 = \sqrt{\frac{2}{fan_{in} + fan_{out}}}$$

Avec fan_{in} et fan_{out} le nombre de neurones d'entrée et de sortie de la couche courante, respectivement.
Dans le cas de la loi uniforme, les poids ω sont initialisés entre $[-lim \dots lim]$ où chaque valeur présente dans l'intervalle est équiprobable et avec

$$lim = \sqrt{\frac{6}{fan_{in} + fan_{out}}}$$

Toujours avec fan_{in} et fan_{out} le nombre de neurones d'entrée et de sortie de la couche courante, respectivement.

2.7.6 Fonction d'erreur

L'objectif d'une fonction d'erreur est de quantifier une erreur entre une vérité terrain y et une prédiction $\hat{y}$. La fonction d'erreur est un choix à définir par l'utilisateur et de nombreuses possibilités sont disponibles dans la littérature. Certaines sont plutôt destinées à être utilisées dans des cas de classifications et d'autres dans des cas de régression. Cette fonction d'erreur permet de quantifier l'erreur entre la sortie attendue y et la prédiction $\hat{y}$. Notée $loss(y, \hat{y})$ il s'agit toujours d'une fonction nulle lorsque $y = \hat{y}$ et qui est de plus en plus élevée lorsque l'écart entre y et $\hat{y}$ augmente.

L'entropie croisée est l'une des plus utilisées pour les cas de classifications multi-classe. L'équation 2.16 détaille le calcul effectué avec K le nombre de classes du problème. Cette fonction fait partie des fonctions d'erreur probabilistes tout comme la fonction d'erreur de poisson via l'équation 2.17 ainsi que l'entropie relative détaillée via l'équation 2.18.

Erreur d'entropie croisée :

$$loss(y, \hat{y}) = \sum_{i=0}^{i=K} -(y_i \times log(\hat{y_i}) + (1 - y_i) \times log(1 - \hat{y_i})) \tag{2.16}$$

Erreur de Poisson :

$$loss(y, \hat{y}) = \hat{y} - y \times log(\hat{y}) \tag{2.17}$$

Erreur divergente de Kullback et Leibler (Entropie relative) :

$$loss(y, \hat{y}) = y \times log(\frac{y}{\hat{y}}) \tag{2.18}$$

Les fonctions d'erreur quadratique ou absolue (cf équation 2.19 et 2.20) ainsi que leurs variantes telles que l'erreur moyenne absolue en pourcentage 2.21 ou encore l'erreur quadratique moyenne logarithmique 2.22 ainsi que la similitude en cosinus 2.23 font parties des fonctions d'erreur de régression.

Erreur quadratique moyenne :

$$loss(y, \hat{y}) = (y - \hat{y})^2 \tag{2.19}$$

Erreur absolue moyenne :

$$loss(y, \hat{y}) = \|y - \hat{y}\| \tag{2.20}$$

Erreur absolue moyenne en pourcentage :

$$loss(y, \hat{y}) = 100 \times \frac{\|y - \hat{y}\|}{y} \tag{2.21}$$

erreur quadratique moyenne logarithmique :

$$loss(y, \hat{y}) = (log(y + 1) - log(\hat{y} + 1))^2 \tag{2.22}$$

Similarité en cosinus :

$$loss(y, \hat{y}) = -\sum_{i=0}^{i=K} (\|y_i\| \times \|\hat{y_i}\|) \tag{2.23}$$

Les fonctions d'erreur de Hinge (2.24) et Hinge quadratique (2.25) font parties des fonctions d'erreur de classification visant à maximiser la marge entre les classes.

Erreur de Hinge :

$$loss(y, \hat{y}) = max(1 - y \times \hat{y}, 0) \tag{2.24}$$

Erreur de Hinge quadratique :

$$loss(y, \hat{y}) = max(1 - y \times \hat{y}, 0)^2 \tag{2.25}$$

Finalement l'utilisateur possède de nombreux outils permettant de quantifier une erreur entre une vérité terrain y et une prédiction $\hat{y}$. De plus, il est possible d'implémenter sa propre fonction d'erreur soit en combinant plusieurs fonctions déjà citées précédemment soit en utilisant la physique associée au problème. L'objectif de l'apprentissage d'un réseau de neurones étant de minimiser l'erreur de prédiction, toute fonction étant croissante avec l'écart entre y et $\hat{y}$ qui augmente est techniquement possible.

2.7.7 Pas d'apprentissage

Le pas d'apprentissage est un hyper-paramètre important à définir puisqu'il va permettre de définir la quantité de modification appliquée lors de la rétro-propagation de l'erreur. Un pas d'apprentissage trop grand risque de ne pas trouver le minimum de la fonction d'erreur voire même de diverger. Au contraire, un pas d'apprentissage petit augmente les chances de trouver le minimum de la fonction d'erreur, mais au détriment d'un temps de calcul plus élevé. De plus si le pas d'apprentissage est trop petit, il est possible que la solution proposée par le modèle converge vers un minimum local de la fonction sans pouvoir en sortir. L'effet de ces deux extrêmes de pas d'apprentissage est visualisé sur la figure 2.27 à gauche et à droite.

L'idéal serait alors d'avoir un pas d'apprentissage adaptatif qui se réduit au cours de l'apprentissage tel que présenté au milieu de la figure 2.27. Le fait de partir d'un grand pas d'apprentissage permet de converger rapidement vers un minimum de la fonction d'erreur, et la réduction de celui-ci au cours de l'apprentissage permet d'affiner la solution.

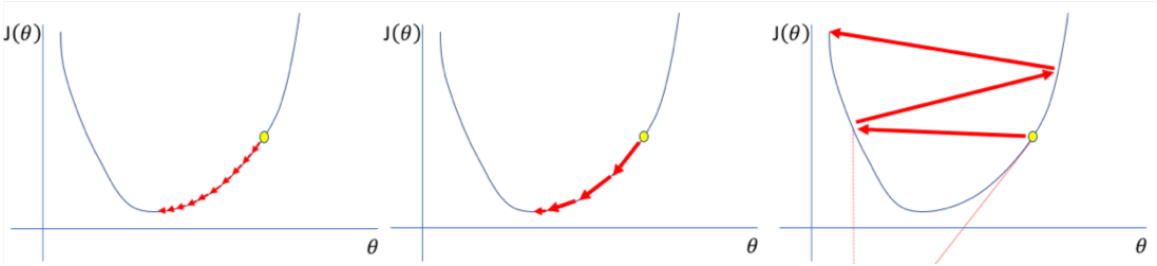


FIGURE 2.27: Effet du pas d'apprentissage pour trouver le minimum de la fonction d'erreur

Plusieurs solutions sont alors disponibles pour l'utilisateur lors de la création du modèle de réseau de neurones. Certains optimiseurs possèdent la capacité de réduire le pas d'apprentissage au fur et à mesure de l'apprentissage, et ainsi de maintenir un pas d'apprentissage adapté tout au long de la phase d'apprentissage.
Il est aussi possible de créer un "callback", autrement dit une fonction qui est appelée à chaque itération, mettant à jour le pas d'apprentissage en fonction de l'itération, voir même de l'erreur actuelle.
Certains "callback" sont déjà implémentés, en particulier celui permettant de réduire le pas d'apprentissage lorsque la fonction d'erreur ne réduit plus pendant plusieurs itérations.

2.7.8 Optimiseur

L'objectif de l'optimiseur est de déterminer comment les paramètres du modèle de réseau de neurones vont être modifiés de façon à minimiser l'erreur de prédiction. Plusieurs choix sont disponibles et déjà implémentés dans la littérature.

La descente de gradient stochastique (SGD) est l'un des optimiseurs les plus connus, mais aussi l'un des plus simples. Un poids ω va être modifié de part le gradient δ_ω et le pas d'apprentissage α tel que décrit dans l'équation 2.26.

$$\omega = \omega - \alpha \times \delta_\omega \tag{2.26}$$

Cet optimiseur possède la possibilité d'ajouter un moment β ainsi qu'un gradient accéléré de Nestronov [71]. L'équation 2.26 devient alors l'équation 2.27 lorsque le paramètre du moment est

positif avec v_{d_ω} la vélocité du poids ω.

$$\begin{cases} v_{d_\omega} = \beta \times v_{d_\omega} - \alpha \times \delta_\omega \\ \omega = \omega + v_{d_\omega} \end{cases} \tag{2.27}$$

L'utilisation du gradient accéléré de Nestronov modifie l'équation 2.26 en

$$\begin{cases} v_{d_\omega} = \beta \times v_{d_\omega} - \alpha \times \delta_\omega \\ \omega = \omega + \beta \times v_{d_\omega} - \alpha \times \delta_\omega \end{cases} \tag{2.28}$$

RMSprop (Root Mean Square propagation) est un autre optimiseur dont l'idée est de maintenir une moyenne mobile (actualisée) du carré des gradients et de diviser le gradient par la racine de cette moyenne. Afin de prévenir le cas où v_{d_ω} est nul ou très proche de 0, un ϵ est ajouté.

$$\begin{cases} v_{d_\omega} = \beta \times v_{d_\omega} + (1-\beta) \times \delta_\omega^2 \\ \omega = \omega - \alpha \times \frac{\delta_\omega}{\sqrt{v_{d_\omega}+\epsilon}} \end{cases} \tag{2.29}$$

Adam est l'un des optimiseurs les plus utilisés pour les réseaux de neurones dont les buts sont de maintenir un pas d'apprentissage adapté en partant du pas d'apprentissage fourni par l'utilisateur, mais aussi d'avoir une mise à jour des poids invariante par rapport à la magnitude du gradient, ce qui est très utile pour traverser des zones avec des gradients très faibles. Finalement, l'optimiseur Adam peut être considéré comme la combinaison de RMSprop et SGD avec momentum ce qui le rend pertinent sur un large éventail de problèmes [72], [73]. L'équation 2.30 détaille la mise à jour des poids utilisée par Adam où m et v sont des moyennes mobiles, $\hat{m}$ et $\hat{v}$ leurs versions avec un biais corrigé et η un correcteur du pas d'apprentissage.

$$\begin{cases} m_t = \beta_1 \times m_{t-1} + (1-\beta_1) \times \delta_{\omega_t} \\ v_t = \beta_2 \times v_{t-1} + (1-\beta_2) \times \delta_{\omega_t}^2 \\ \hat{m}_t = \frac{m_t}{1-\beta_1} \\ \hat{v}_t = \frac{v_t}{1-\beta_2} \\ \omega_t = \omega_{t-1} \times \eta \frac{\hat{m}_t}{\sqrt{v_t}+\epsilon} \end{cases} \tag{2.30}$$

D'autres variantes de l'optimiseur Adam sont disponibles telles que Adagrad qui utilise en plus la fréquence de mise à jour des poids durant l'apprentissage [74], Adadelta qui s'appuie sur une fenêtre glissante des gradients précédents plutôt que d'accumuler tous les gradients passés [75], Adamax qui n'utilise le moment que sur une tranche de valeurs spécifiques [76], et NAdam qui incorpore le moment Nestronov à Adam [77].

2.8 Modèle monomodal et multimodal

Les modèles mono-modaux correspondent aux modèles n'utilisant qu'une seule entrée. La majorité des modèles présents dans la littérature sont monomodaux. Cependant les réseaux de neurones ont la possibilité de s'entrainer sur des données multimodales. Cela correspond au cas où une donnée d'entrée est composée de plusieurs informations de type différent.

Cette possibilité est offerte aux réseaux de neurones via les couches de fusion. De nombreuses couches de fusion sont déjà disponibles dans la littérature, telles que la concaténation, l'addition et la multiplication. Cependant l'utilisateur a la possibilité de créer sa propre couche de fusion telle

qu'une concaténation pondérée, avec le poids de chaque modalité appris lors de la phase d'apprentissage.

L'utilisateur doit alors choisir à quel moment fusionner les sorties de deux couches. La fusion précoce ou tardive sont les plus utilisées mais il existe des fusions hybrides.
La fusion précoce se place au début du modèle et a l'avantage d'avoir un modèle plus simple et moins gourmand en capacité de calcul de la machine. Cependant les données sont mélangées dès la couche de fusion et rend l'interprétation du modèle plus difficile, ce qui n'aide pas à démystifier l'effet boîte noire du réseau de neurones.
Au contraire une fusion tardive se positionne vers la fin d'un modèle. Chaque partie des données est alors traitée indépendamment avant d'être rassemblée à la fin du modèle par la couche de fusion. Ceci rend l'interprétation du modèle plus aisée, mais augmente considérablement le temps de calcul et les capacités nécessaires de la machine de calcul.

Il est finalement possible d'imaginer un nombre indéfini de possibilité de modèles multimodaux ; par exemple plusieurs images prises de différents points de vue d'une scène 3-dimensions.
Dans des cas appliqués au domaine médical, il est possible d'avoir plusieurs types d'images en entrée en combinant par exemple une IRM et un scanner.
Il est aussi possible de combiner une image médicale avec un vecteur de biomarqueurs du patient.
Une autre possibilité serait de combiner plusieurs biomarqueurs dérivés d'une IRM et ainsi fournir de l'imagerie structurelle, de diffusion et fonctionnelle à un modèle.

La figure 2.28 présente un exemple d'architecture multimodale dont une donnée est composée de plusieurs informations. La figure utilise ensuite des perceptrons multi couches mais il est possible d'utiliser des réseaux de neurones convolutifs ou encore des combinaisons des deux.

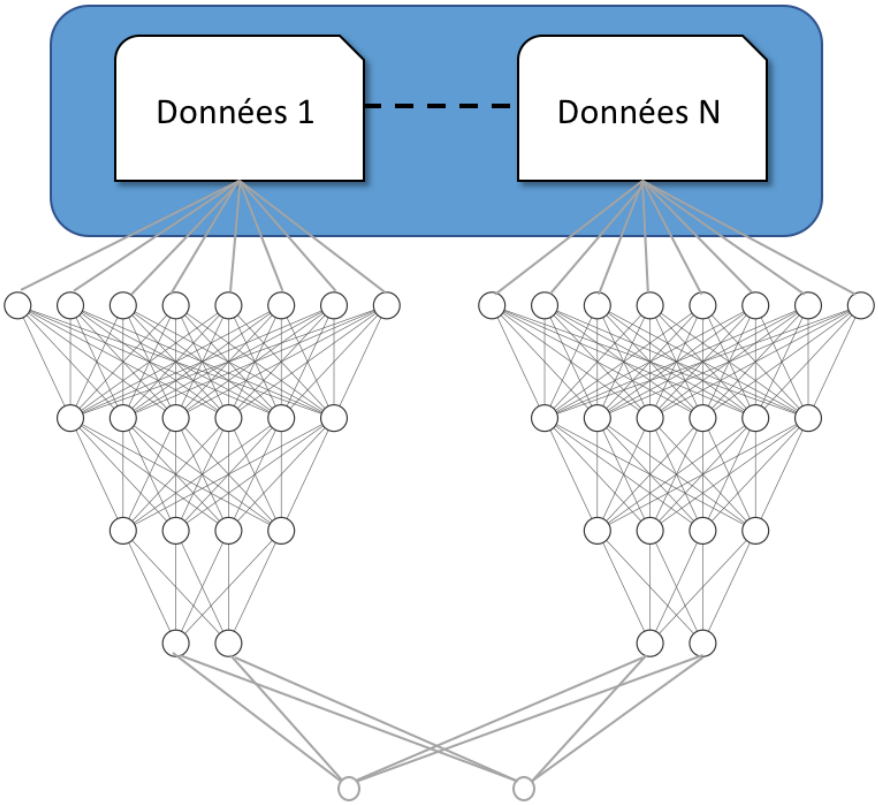


FIGURE 2.28: Exemple d'architecture de réseau de neurones multimodal

2.9 entraînement d'un modèle

Toutes les sections du chapitre 2 permettent de créer un réseau de neurones artificiels adapté à différentes applications. Cette section présente l'entraînement d'un modèle avec un algorithme simplifié 1.

La première étape (ligne 1 de l'algorithme 1) consiste à préparer les données afin qu'elles soient utilisables par un réseau de neurones. Dans le cas d'un réseau de neurones entièrement connectés, cette étape est réduite à la simple action d'obtenir les données sous forme d'un vecteur, de façon à ce que la n^{eme} donnée puisse être accessible par $data[n]$. Pour un réseau de neurones convolutifs, les sorties des couches de convolution étant stockées sur les "channels", il est nécessaire d'ajouter une dimension pour les données ne possédant qu'un seul "channel". Finalement, après exécution de cette ligne les données doivent être dans l'un des formats suivants en fonction du type de donnée :

- (nb_data, x) : vecteur de dimension x
- $(nb\_data, x, y, 1)$: image mono-channel de dimension (x, y)
- $(nb\_data, x, y, channel)$: image multi-channel de dimension (x, y)
- $(nb\_data, x, y, z, 1)$: volume mono-channel de dimension (x, y, z)
- $(nb\_data, x, y, z, channel)$: volume multi-channel de dimension (x, y, z)
- ...

La ligne 2 de l'algorithme 1 consiste à séparer de façon aléatoire les données de façon à obtenir un jeu de données d'entraînement d'environ 70% et un jeu de données de test d'environ 30%.

Les lignes 3 et 4 de l'algorithme 1 permettent d'appliquer un pré traitement aux données tel qu'une normalisation ou une standardisation. Il est nécessaire d'appliquer le même pré traitement aux données d'entraînement et de test.

La ligne 5 de l'algorithme 1 crée le modèle défini par l'utilisateur.

Les lignes 6 et 7 de l'algorithme 1 sont des hyper-paramètres du modèle avec le pas d'apprentissage et l'optimiseur, ainsi que le choix de la fonction d'erreur et la métrique utilisée.

La ligne 8 de l'algorithme 1 compile le modèle avec les hyper-paramètres choisis par l'utilisateur.

Les lignes 9 et 10 de l'algorithme 1 permettent de définir les valeurs des variables $batch\_size$ et $epochs$ correspondant respectivement au nombre de données qui vont traverser le modèle avant que l'erreur ne soit rétropropagée et que les poids soient mis à jour, et le nombre de fois que le jeu de données d'entraînement va traverser le modèle avant que l'apprentissage s'arrête.

La ligne 11 de l'algorithme 1 lance la boucle d'apprentissage du modèle et la ligne 12 calcule les prédictions sur les données de test de façon à obtenir le comportement du modèle sur des données qu'il n'a jamais vues durant l'entrainement. La variable $historic$ contient toutes les informations collectées au cours de l'apprentissage et permet par exemple d'afficher les courbes d'apprentissage.

Algorithm 1 Algorithme complet d'apprentissage d'un réseau de neurones

1: $data \leftarrow reshape(dataset)$
2: $XTrain, XTest, YTrain, YTest \leftarrow split(data, groundtruth)$
3: $XTrain \leftarrow pre\_processing(XTrain)$
4: $XTest \leftarrow pre\_processing(XTest)$
5: $model \leftarrow create\_model()$
6: $optimizer \leftarrow optimizer(1e-3)$
7: $loss \leftarrow loss()$
8: $model.compile(optimizer, loss)$
9: $batch\_size \leftarrow 32$
10: $epochs \leftarrow 100$
11: $historic \leftarrow model.fit(XTrain, YTrain, batch\_size, epochs)$
12: $model.predict(XTest)$

CHAPITRE 3

Outils de visualisation et d'interprétation d'un réseau de neurones convolutifs

Les réseaux de neurones sont de plus en plus utilisés dans de nombreux domaines et pour des applications très variées. Cependant, bien que les résultats soient prometteurs les réseaux de neurones sont à l'heure actuelle toujours considérés comme des boites noires. En effet, la communauté scientifique accepte et admet que ces modèles d'intelligence artificielle profonds fonctionnent et produisent d'excellents résultats, cependant l'étape de leur interprétation reste difficile.

Le chapitre 3 détaille les méthodes de visualisation et d'interprétation d'un réseau de neurones convolutifs. La section 3.1 présente la méthode d'occlusion partielle de l'entrée et la section 3.2 les "saliency maps" (ou carte de pertinence). Les sections 3.3 et 3.4 détaillent les "class activation maps". La section 3.5 présente quant à elle la méthode développée durant ces travaux de thèse, nommée "CNN eyes visions".

Chacune de ces sections commencera par une présentation de la méthode ainsi que leurs avantages et inconvénients, puis des exemples de résultats obtenus avec ces différentes méthodes de visualisation seront présentés.

La section 3.6 compare les résultats obtenus avec la méthode développée durant cette thèse et les autres méthodes de visualisation déjà présentes dans l'état de l'art.
La section 3.7 présente un logiciel de visualisation développé durant ces travaux de thèse.

Le chapitre se termine par la section 3.8 qui fait le bilan des différentes méthodes de visualisation des zones discriminantes.

3.1 Occlusion partielle de l'entrée

3.1.1 Méthodologie

La méthode de visualisation des zones discriminantes par occlusion partielle de l'entrée s'appuie sur les différences entre les prédictions sur l'image entière et cette même image à laquelle une partie a été cachée. Ainsi en parcourant l'image avec un filtre qui supprime une partie des données de l'entrée, il est possible de retrouver les zones importantes dans la prédiction pour le modèle, dans un but de prédiction. La figure 3.1 montre un exemple d'utilisation de cette méthode. La partie haute de la figure 3.1 calcule les prédictions sur la totalité de l'image avec P_i la probabilité d'appartenance à la classe "chat". La partie basse de la figure 3.1 calcule cette fois-ci les prédictions avec une partie de l'image cachée, et P_i' est la nouvelle probabilité d'appartenance à la classe "chat" par le même modèle.
Si $P_i \simeq P_i'$ alors la probabilité d'appartenance à la classe "chat" ne change pas vraiment en ayant ou non l'information présente dans la partie cachée. Cette zone est alors considérée comme non discriminante.
Inversement si $P_i >> P_i'$ alors la probabilité d'appartenance à la classe "chat" a diminué en cachant la partie de l'entrée. La zone cachée est alors considérée comme discriminante.

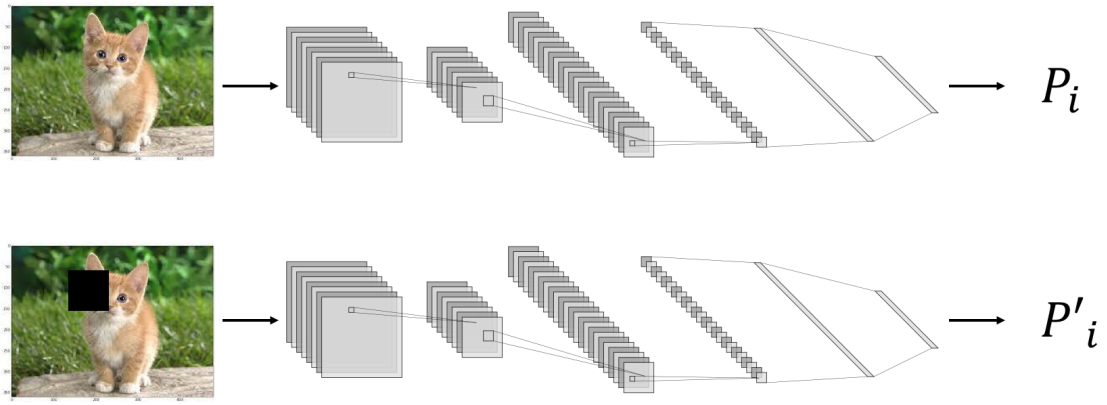


FIGURE 3.1: Détail de la visualisation des voxels discriminant avec la méthode d'occlusion partielle de l'entrée

En pratique, cette méthode utilise plusieurs zones cachées permettant finalement de recouvrir toute l'image et attribue un niveau d'importance à la zone courante en calculant la différence $P_i - P_i'$. Le principe derrière cette méthode est relativement simple, mais les temps de calcul sont longs. De plus, il reste à la charge de l'utilisateur de trouver une taille de filtre d'occlusion adéquate.

3.1.2 Visualisations avec l'occlusion partielle de l'entrée

La figure 3.2 présente la position du cervelet dans le cerveau. En effet une anormalité a été induite par simulation dans le cervelet et la zone discriminante à retrouver par la méthode de visualisation par occlusion partielle de l'entrée est donc le cervelet.

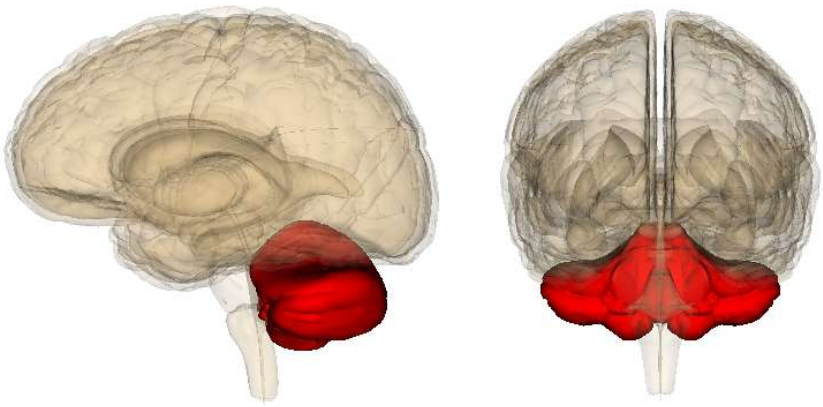


FIGURE 3.2: Position du cervelet dans le cerveau

La figure 3.3 montre le résultats obtenus via la méthode d'occlusion partielle de l'entrée. Le cervelet est bien ciblé, mais plus particulièrement la partie haute du cervelet alors que la basse est omise.

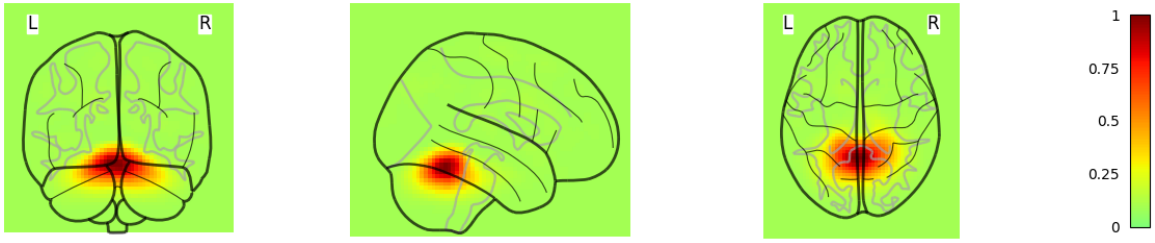


FIGURE 3.3: Exemple de visualisation des voxels discriminant avec la méthode d'occlusion partielle de l'entrée sur des données IRM 3D du cerveau avec une anomalie dans le cervelet

3.2 Saliency map

3.2.1 Méthodologie

Les "saliency maps" (ou carte de pertinences) sont l'une des méthodes de visualisation du fonctionnement d'un réseau de neurones convolutifs introduite par Simonyan et al. en 2014 [78]. L'idée à l'origine des "saliency maps" repose sur l'analyse d'un problème inverse. En effet cette méthode pourrait être décrite par la question suivante :
Quels sont les pixels / voxels pour lesquels une petite variation implique une grande variation dans la prédiction.
La réponse à cette question permet de retrouver les pixels les plus discriminants dans une donnée.

D'autres travaux donnent des améliorations de ces "saliency maps" tels que ceux de Mundhenk et al. en 2020 [79] où les "saliency maps" de plusieurs couches sont combinées afin d'obtenir une visualisation plus pertinente.

Quelle que soit la méthode basée sur les saliency maps, l'équation 3.1 est toujours à la base des résultats produits, avec *output* la prédiction, *input* l'image d'entrée et S la saliency map.

$$S = \frac{\delta_{output}}{\delta_{input}} \tag{3.1}$$

3.2.2 Visualisations avec saliency map

Les figures 3.4 et 3.5 sont issues des articles respectifs de Simonyan et al. en 2014 [78] et Mundhenk et al. en 2020 [79] ; elles présentent des visualisations obtenues avec la méthode saliency map sur le jeu de données Image-net [17].

La figure 3.4 montre que la zone où se situe le chien dans l'image est mise en évidence par la méthode de visualisation.

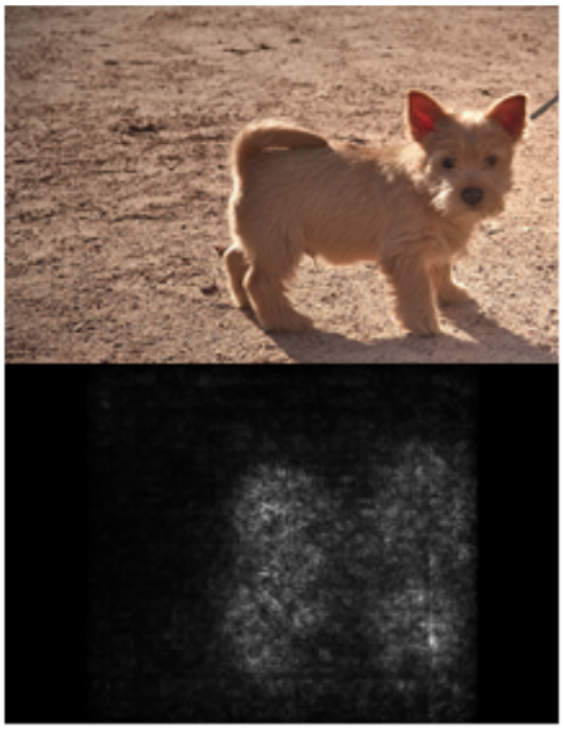


FIGURE 3.4: Visualisation des voxels discriminants (bas) de la classe "chien" pour l'image d'entrée (haut) avec la méthode saliency map [78]

La figure 3.5 quant à elle montre les "saliency maps" de chaque couche de convolution du modèle. Il est alors possible de constater que les saliency maps plus proches de l'entrée possèdent une meilleure résolution spatiale que celles obtenues en profondeur. Cependant, la fusion de toutes ces "saliency maps" permet d'obtenir une visualisation plus précise de l'objet en question.

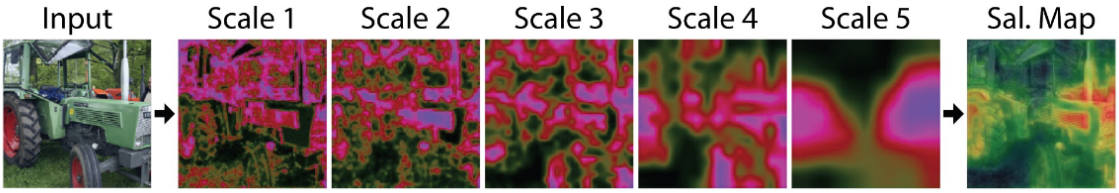


FIGURE 3.5: Visualisation des voxels discriminants (droite) de la classe "tracteur" pour l'image d'entrée (gauche) avec la méthode saliency map fusionnée [79], avec les saliency maps intermédiaires

3.3 Class Activation Mapping (CAM)

3.3.1 Méthodologie

La Class Activation Map (CAM) est une autre méthode de visualisation des zones discriminantes dans une donnée. Celle-ci s'appuie sur les décompositions obtenues sur la dernière couche de convolution et les poids de la couche entièrement connectée qui la suit.

En appliquant une opération de "Global Average Pooling" (GAP) dans le modèle, ce qui permet de ne conserver qu'une valeur moyenne pour chaque décomposition, puis une couche entièrement connectée avant celle de prédiction il est possible de calculer la somme des décompositions multipliées par leur poids de la couche entièrement connectée menant à une certaine classe [80]. L'équation 3.2 et la figure 3.6 détaillent la méthode de visualisation CAM, avec V le résultat obtenu, d_i la décomposition i de la dernière couche de convolution contenant N filtres, et $\omega_{i,k}$ le poids du lien de la couche dense entre la décomposition i et la classe k.

$$V = \sum_{i=0}^{i=N} \omega_{i,k} \times d_i \tag{3.2}$$

3.3.2 Visualisations avec CAM

La figure 3.7 montre un exemple de visualisation avec la méthode Class Activation Map pour un modèle utilisant un "Global Average Pooling" entrainé sur le jeu de données Image-net [17]. L'avantage de ce type de méthode est qu'il est possible de cibler les zones discriminantes d'une donnée en fonction de la classe souhaitée. Autrement dit, pour les données multi-classe, la visualisation obtenue via la méthode CAM sera différente selon la classe. Par exemple sur la figure 3.7, la visualisation CAM de l'image de droite cible l'homme en train de couper l'arbre, mais il serait possible de cibler l'arbre en changeant la classe à visualiser (le k dans l'équation 3.2). Cependant, afin de pouvoir appliquer la méthode CAM, il faut utiliser un modèle contenant un "Global Average Pooling" ce qui limite le nombre d'architectures sur lesquelles la méthode peut être appliquée.

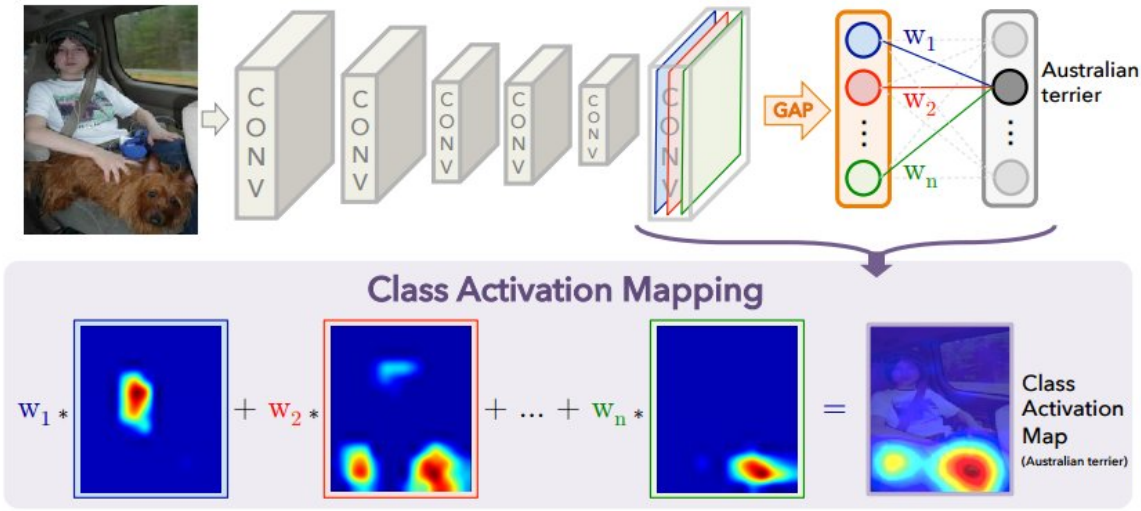


FIGURE 3.6: Détail de la méthode de visualisation des voxels discriminants CAM

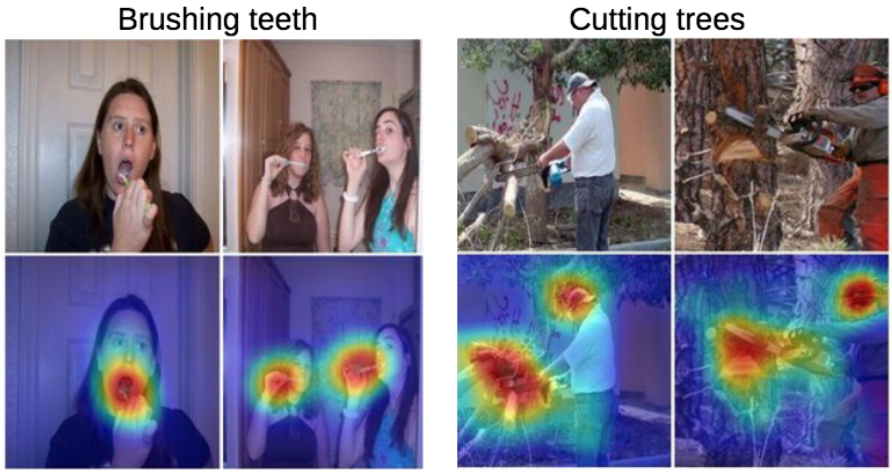


FIGURE 3.7: Exemple de visualisation de la méthode CAM pour différentes classes d'Image-net [17] (figure issue de [80])

3.4 Gradient weighted Class Activation Mapping (gradCAM)

3.4.1 Méthodologie

Gradient weighted Class Activation Mapping (gradCAM) est la méthode la plus utilisée dans l'état de l'art pour la visualisation des zones discriminantes dans une donnée. Cette méthode est une extension des "Class Activation Map" (voir section 3.3) et permet d'être employée sur tout type de modèle, autrement dit les modèles sans "Global Average Pooling", et sur plusieurs types

de tâche (classification, annotation et réponse aux questions dans une image). GradCAM utilise la rétro-propagation de l'erreur jusqu'à une certaine couche afin d'obtenir les "features maps" (ou cartes de caractèristiques) [81]. La figure 3.8 schématise les opérations de la méthode gradCAM.
La première étape consiste à calculer $\frac{\delta y^c}{\delta A^k}$ le gradient de la classe c rétro propagé jusque la couche A pour chacun des filtres k.
Ensuite ces $\frac{\delta y^c}{\delta A^k}$ sont moyennés de façon à obtenir une valeur α_k^c, ce qui correspond au "global average pooling" de la méthode CAM. Pour une image de dimension $i \times j$ cela correspond au calcul suivant :
$\alpha_k^c = \frac{1}{i \times j} \sum_i \sum_j \frac{\delta y^c}{\delta A_{i,j}^k}$
Finalement, la visualisation est calculée par combinaison linéaire des α_k^c et A^k comme l'indique l'équation 3.3.

$$\begin{cases} gradCAM^c = \sum_k \alpha_k^c \times A^k \\ gradCAM^c = \sum_k \frac{1}{i \times j} \sum_i \sum_j \frac{\delta y^c}{\delta A_{i,j}^k} \times A_{i,j}^k \end{cases} \tag{3.3}$$

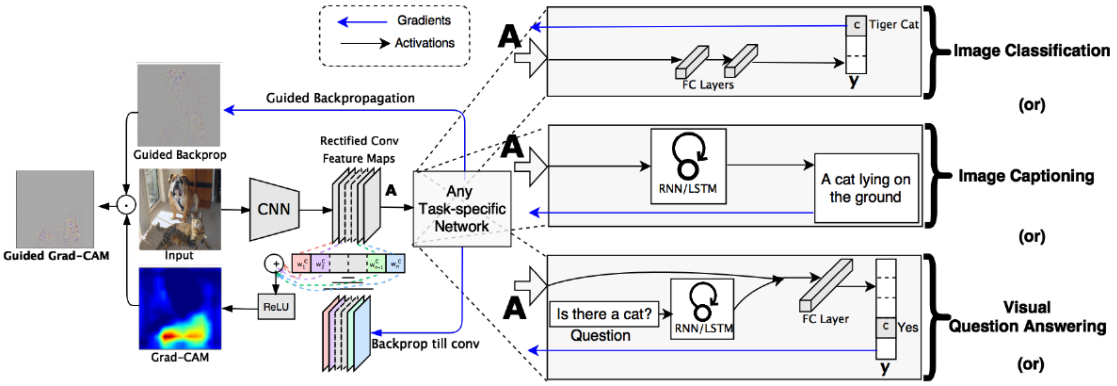


FIGURE 3.8: Détail de la méthode de visualisation des voxels discriminants GradCAM [81]

Depuis les travaux de Ramprasaath et al. en 2017 [81], Chattopadhay et. al en 2018 [82] et Omeiza et al. en 2019 [83] ont amélioré cette méthode avec gradCAM++ et smooth gradCAM++ [84]. La figure 3.9 détaille la méthode gradCAM++ et l'équation 3.4 son calcul.

$$\begin{cases} gradCAM++^c = relu(gradCAM^c) \\ gradCAM++^c = relu(\sum_k \alpha_k^c \times A^k) \\ gradCAM++^c = relu(\sum_k \frac{1}{i \times j} \sum_i \sum_j \frac{\delta y^c}{\delta A_{i,j}^k} \times A_{i,j}^k) \end{cases} \tag{3.4}$$

Étant une amélioration de la méthode CAM, gradCAM permet aussi de visualiser les zones discriminantes pour des données multi-classes. Cependant, la partie de la rétro-propagation de l'erreur peut consommer un temps de calcul conséquent en fonction de l'architecture du modèle et la couche de convolution que l'utilisateur souhaite visualiser.

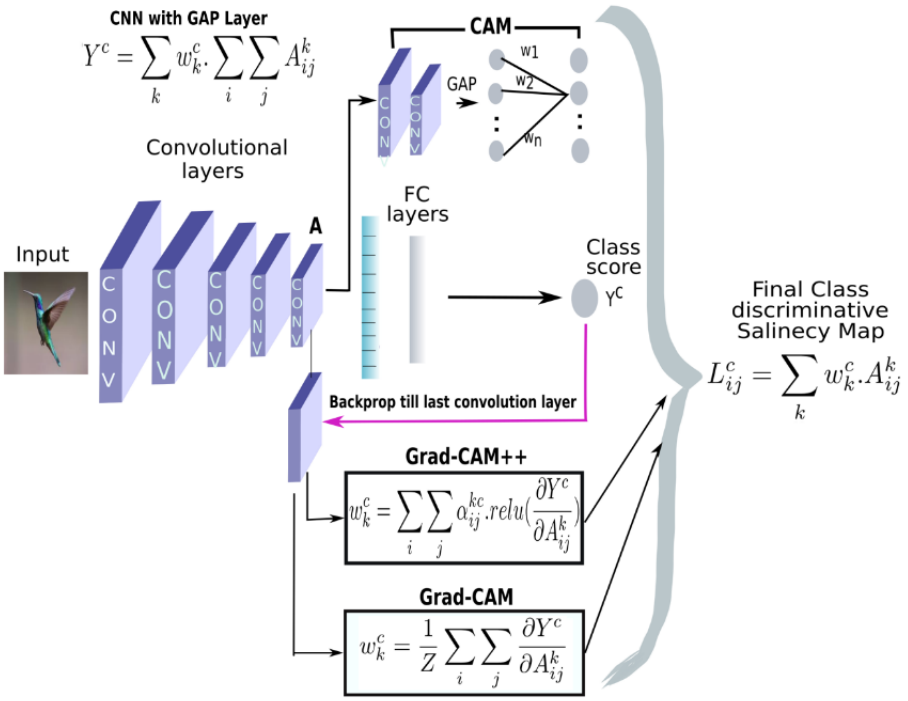


FIGURE 3.9: Détail de la méthode de visualisation des voxels discriminants GradCAM++ [83]

3.4.2 Visualisations avec gradCAM

La figure 3.10 présente un exemple de visualisation obtenue avec la méthode gradCAM sur une donnée issue du jeu de données Image-net [17]. La visualisation de données multi-classe est alors clairement visible, puisqu'en visualisant les zones discriminantes de l'image pour la classe "chien", celui-ci est bien ciblé par gradCAM et il en est de même pour la classe "chat".

La figure 3.11 quant à elle montre des résultats et compare les visualisations de différentes images provenant du jeu de données Image-net [17] avec les méthodes gradCAM, gradCAM++ et smooth gradCAM++. La figure 3.11 montre l'intérêt des améliorations effectuées par Chattopadhay et. al en 2018 [82] et Omeiza et al. en 2019 [83] avec des visualisations plus lissées et englobant une zone discriminante plus proche de ce que l'utilisateur pourrait attendre.

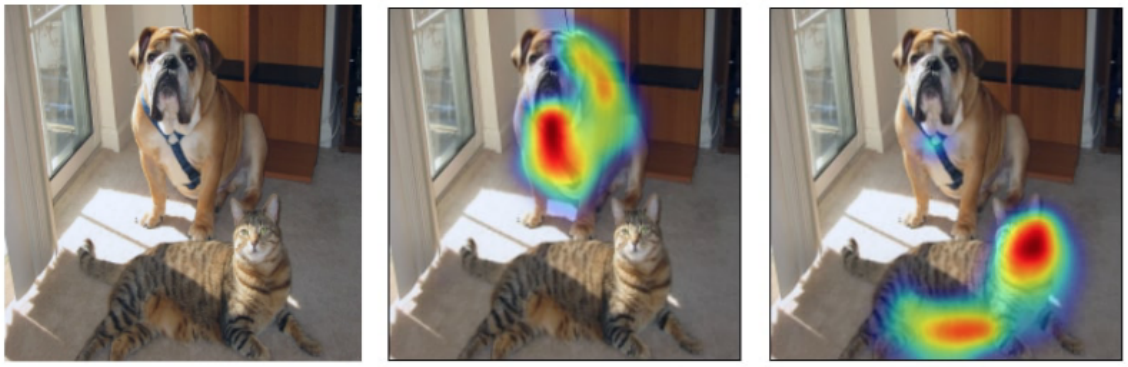


FIGURE 3.10: Exemple de visualisation de la méthode gradCAM pour une image provenant d'Image-net [17] contenant un chien et un chat, avec l'image originale à gauche, la visualisation de la classe chien au milieu, et la visualisation de la classe chat à droite (figure issue de [81])

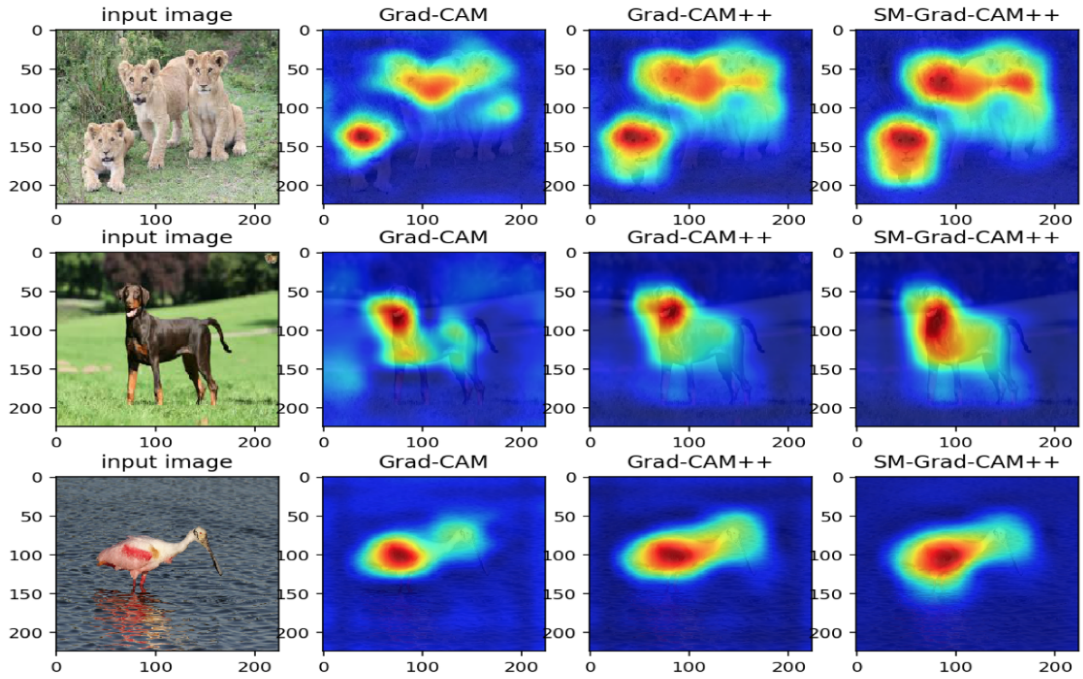


FIGURE 3.11: Comparaison des visualisations des méthodes gradCAM, gradCAM++ et smooth-gradCAM++ pour différentes classes d'Image-net [17] (figure issue de [83])

3.5 CNN eyes visions

Les principales méthodes de visualisation des zones discriminantes d'un réseau de neurones présentes dans l'état de l'art ont été abordées dans les sections 3.1, 3.2, 3.3 et 3.4 avec leurs avantages et inconvénients. Cependant, ces méthodes peuvent présenter des inconvénients lorsque l'utilisateur

souhaite les appliquer sur tout type d'architecture et en particulier sur les modèles 3-dimensions. C'est pourquoi une méthode alternative a été développée lors de ces travaux de thèse.

Si l'occlusion partielle de l'entrée cherche à retrouver les zones discriminantes en cachant une partie de l'entrée, les "saliency maps" les retrouvent en cherchant les zones sur lesquelles une petite variation implique une grande variation sur les prédictions ainsi que CAM et gradCAM via la rétropropagation de l'erreur de prédiction. La méthode développée ici, nommée "CNN eyes visions", cherche les zones importantes dans une donnée en fusionnant les décompositions intermédiaires d'un modèle. Autrement dit, CNN eyes visions tente de retrouver ce que "regarde" un modèle lorsqu'une entrée lui est fournie. Le nom de la méthode correspond au fait qu'elle tente de prendre les yeux du modèle pour "voir" où "regarde" un modèle.

3.5.1 Méthodologie

Afin de retrouver les zones considérées comme importantes par un modèle, la méthode nommée CNN eyes visions extrait toutes les décompositions intermédiaires d'un modèle. Ainsi pour chaque couche de convolution composée de N filtres, N visualisations intermédiaires sont extraites. Ces N visualisations subissent un seuillage afin de supprimer les valeurs négatives afin de ne conserver uniquement que les activations positives. Elles sont ensuite interpolées vers la dimension de l'entrée de façon à régler le souci des décompositions ayant subi une opération de "pooling" ou éventuellement une convolution dite "valide" où la dimension de la sortie est réduite par rapport à l'entrée en fonction de la taille du filtre de convolution. Finalement ces visualisations sont moyennées dans le but d'obtenir la visualisation moyenne de la couche courante.
Ceci permet d'obtenir une visualisation représentant les zones où "regarde" le modèle sur la couche courante lorsqu'on lui fournit une entrée.

Cette opération peut être réitérée sur toutes les couches de convolution du modèle afin d'obtenir la visualisation de chaque couche de convolution. Ensuite, il est possible de fusionner toutes ces visualisations par couche de convolution et de normaliser le résultat afin d'obtenir une carte d'activation considérée comme "où regarde le modèle sur cette donnée" sur la totalité du modèle.

La figure 3.12 et l'algorithme 2 détaillent la méthode CNN eyes visions permettant d'obtenir une visualisation par donnée.

Cette méthode permet d'obtenir des résultats intéressants comme le montre les figures de la section 3.5.2, cependant il est possible d'aller plus loin dans certains cas.

En effet, en citant l'exemple des données MNIST [85] (autrement dit les images de chiffres manuscrits), le chiffre est centré dans l'image et il est alors possible de calculer la visualisation moyenne d'un chiffre. Plus particulièrement, les données d'imagerie médicale sont souvent recalées afin que les données des patients soient comparables entre elles. C'est le cas pour les données d'IRM du cerveau dans les templates MNI (voir section 1.2.3). Ceci permet de calculer la visualisation moyenne d'une classe et plus précisément la visualisation moyenne de la classe "patient atteint de la maladie" et celle de la classe "sujet sain". Ainsi la méthode CNN eyes visions permet de savoir où "regarde" particulièrement sur une classe et l'autre. Finalement il est possible de calculer la différence absolue entre les visualisations de chaque classe de façon à retrouver les zones discriminantes puisque le résultat obtenu lors du calcul de la différence absolue permet d'obtenir une visualisation qui représente les zones où un modèle "regarde" pour une classe mais pas pour l'autre.

Cette méthode de visualisation a été validée sur des données simulées [86] et les résultats obtenus

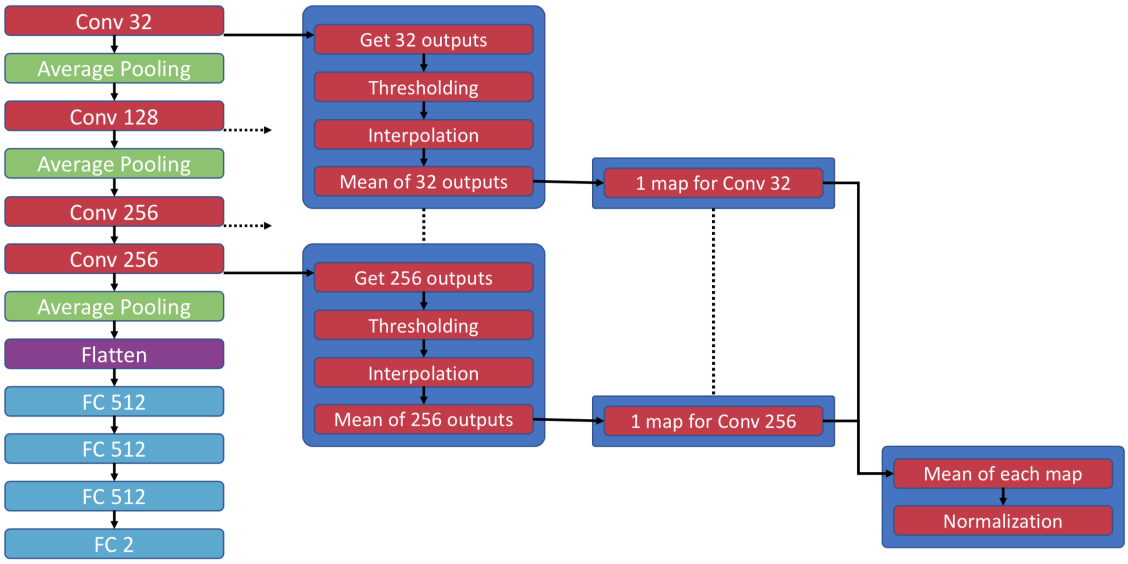


FIGURE 3.12: Détail de la méthode de visualisation des voxels discriminants CNN eyes visions

seront présentés dans le chapitre suivant section 4.3.

3.5.2 Visualisations avec CNN eyes visions

La figure 3.13 présente un exemple de visualisation des zones importantes pour le jeu de données MNIST [85]. Les chiffres manuscrits étant centrés, il est possible de calculer la visualisation moyenne de chacun des chiffres présents dans la partie de test du jeu de données. Il est alors possible de constater que le modèle "regarde" particulièrement certaines zones du contour des chiffres pour les distinguer.

D'autres figures de visualisation utilisant la méthode CNN eyes visions sont présentées dans la section 3.6 qui vise à comparer cette méthode à gradCAM, la méthode la plus couramment utilisée dans la littérature.

Algorithm 2 Calcul des visualisations via la méthode CNN eyes visions

$model \leftarrow$ `charger un modèle`
$visualisation\_sain \leftarrow zeros(model.input\_shape)$
$visualisation\_patient \leftarrow zeros(model.input\_shape)$
for $d \in$ `jeu de données de test` **do**
    $ret \leftarrow$ `Calcul de la visualisation de` d
    if $d \in$ `classe sain` **then**
        $visualisation\_sain \leftarrow visualisation\_sain + ret$
    end if
    if $d \in$ `classe patient` **then**
        $visualisation\_patient \leftarrow visualisation\_patient + ret$
    end if
end for
$visualisation\_sain \leftarrow mean(visualisation\_sain)$
$visualisation\_patient \leftarrow mean(visualisation\_patient)$
$visualisation\_finale \leftarrow abs(visualisation\_sain - visualisation\_patient)$

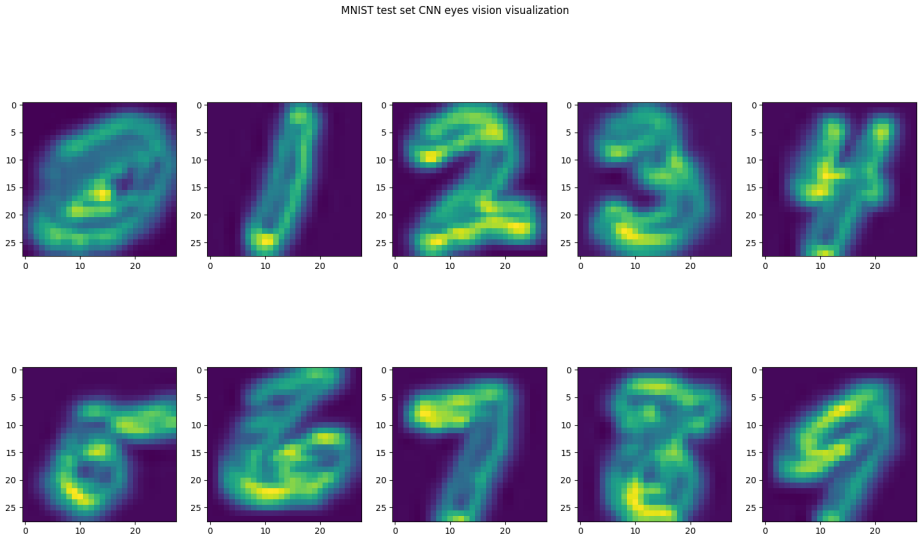


FIGURE 3.13: Visualisation moyenne des voxels discriminants de chaque classe du jeu de données MNIST avec la méthode CNN eyes visions

3.6 Comparaisons des visualisations

Afin de comparer la méthode développée durant ces travaux de thèse et gradCAM, la méthode la plus couramment utilisée dans l'état de l'art, plusieurs figures de visualisation ont été produites.

3.6.1 Transfert learning et jeu de données simulées

Un jeu de données simple a été simulé de façon à obtenir une image 2-dimensions en disposant de la vérité terrain avec la position des zones discriminantes. Ensuite les modèles déjà entrainés sur Image-net [17] ont été utilisés via le "transfer learning" (voir la section 2.4) de façon à discriminer le jeu de données simulées. Ainsi, le jeu de données est composé de 512 images de dimensions $224 \times 224 \times 3$ pixels contenant une forme de taille et position aléatoires. La génération de ce jeu de données vérifie que chaque forme est entièrement contenue dans l'image et que sa taille ne dépasse pas la moitié de celle-ci. Deux classes équilibrées de 256 images chacune sont générées, à savoir la classe "cercle" et la classe "carrée". La forme est remplie de pixels gris, avec un tuple $(128, 128, 128)$ et le fond est nul via le tuple $(0, 0, 0)$.

Les modèles VGG [18], ResNet [19] et GoogleNet [20] sont ensuite ré-entrainés sur ces données puis les visualisations via les méthodes gradCAM et CNN eyes visions sont calculées. Les figures 3.14, 3.15 et 3.16 présentent les résultats obtenus pour 5 données et pour les trois modèles, avec l'image d'entrée en haut, les visualisations obtenues via CNN eyes visions au milieu et celle via gradCAM en bas.

La figure 3.14 montre les résultats de visualisation obtenus pour le modèle VGG [18]. Alors que gradCAM met en évidence la totalité de la forme correspondant à la classe, la méthode CNN eyes visions quant à elle se concentre sur les contours de la forme.

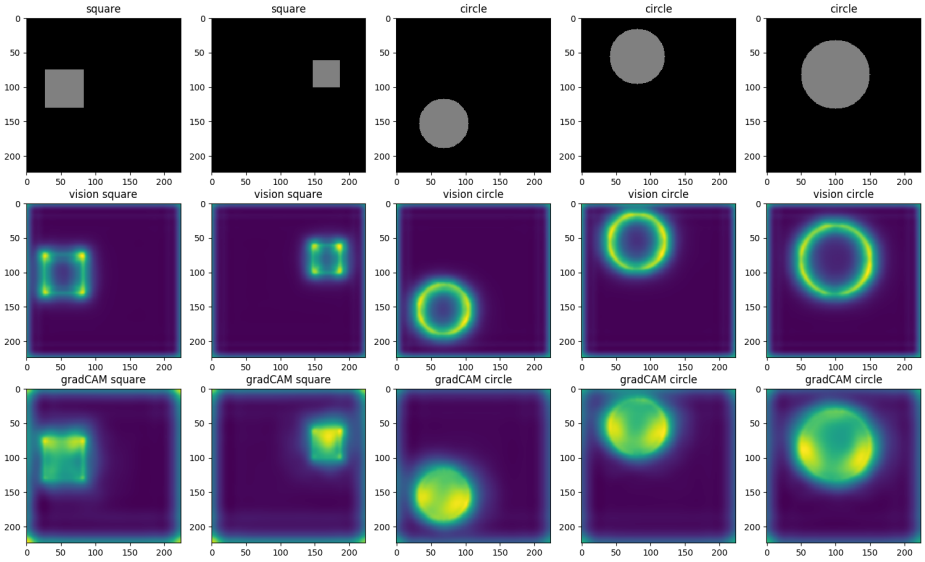


FIGURE 3.14: Comparaison des visualisations obtenues avec GradCAM et CNN eyes visions sur un jeu de données simulées pour le modèle de transfer learning basé sur VGG

La figure 3.15 présente les visualisations obtenues pour le modèle ResNet [19]. Celle-ci sont très

peu intenses, voire même invisibles sans appliquer un changement d'échelle, que ce soit pour la méthode gradCAM ou CNN eyes visions.

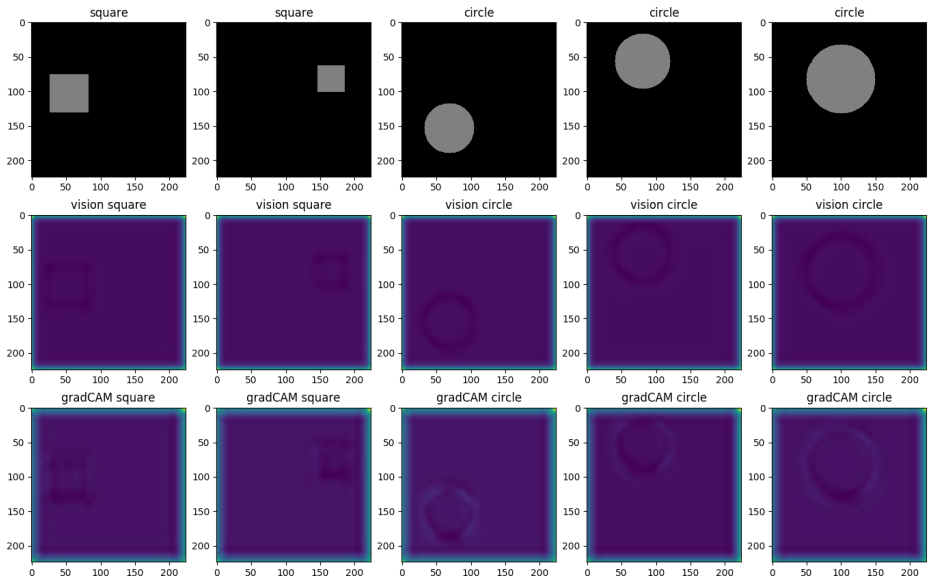


FIGURE 3.15: Comparaison des visualisations obtenues avec GradCAM et CNN eyes visions sur un jeu de données simulées pour le modèle de transfer learning basé sur ResNet

La figure 3.16 montre les visualisations obtenues pour le modèle GoogleNet [20]. Pour les deux méthodes, la visualisation se concentre sur les contours de la forme, bien que la méthode gradCAM fournit des résultats plus intenses.

La table 3.1 détaille les scores d'exactitude obtenus pour les trois modèles précédents sur le jeu de données simulées contenant des formes (cercle ou carré). Il est alors possible de constater que le modèle VGG qui a les meilleures visualisations obtient aussi les meilleurs scores d'exactitude. Ces scores sont suivis de près par le modèle GoogleNet qui obtient aussi de très bonnes visualisations des zones discriminantes. Le modèle ResNet quant à lui ne parvient pas à discriminer les deux formes et les visualisations montrent des zones discriminantes très peu intenses.

	VGG	ResNet	GoogleNet
Exactitude train	100.0%	50.0%	97.3%
Exactitude test	100.0%	50.0%	100.0%

TABLE 3.1: Exactitude sur le jeu de données d'entrainement et de test pour les trois modèles

Sur ce jeu de données simulées utilisant le transfer learning, les visualisations obtenues via gradCAM et CNN eyes visions sont très similaires et comparables. Cependant, la méthode CNN eyes visions présente un avantage non négligeable. En effet, cette méthode n'utilisant pas les couches de

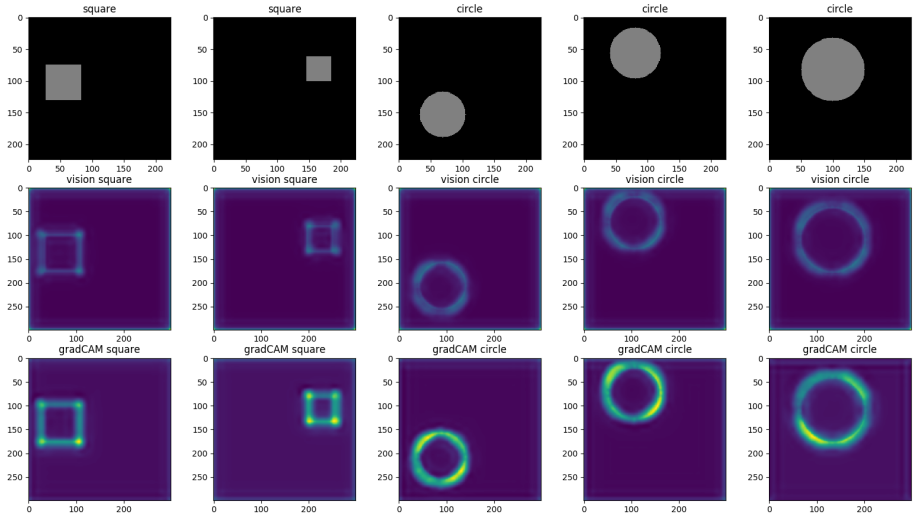


FIGURE 3.16: Comparaison des visualisations obtenues avec GradCAM et CNN eyes visions sur un jeu de données simulées pour le modèle de transfer learning basé sur GoogleNet

prédiction, il est possible de calculer les visualisations avant d'effectuer le transfer learning. GradCAM quant à lui utilise les prédictions pour calculer les visualisations des zones discriminantes, il est donc nécessaire d'entrainer le modèle avant de calculer ces visualisations.

Ceci donne l'avantage à la méthode CNN eyes visions dans le cas du transfert learning puisque l'utilisateur peut calculer les visualisations avant d'entrainer le modèle. Ainsi, en utilisant les visualisations et l'information a priori sur le jeu de données, l'utilisateur peut choisir l'architecture du modèle la plus adaptée au problème. En effet, les modèles VGG et GoogleNet obtiennent les meilleures visualisations et les meilleurs scores, tandis que le modèle ResNet obtient de mauvaises visualisations et ne parvient pas à discriminer les deux formes. L'utilisateur se serait donc naturellement dirigé vers le modèle VGG ou GoogleNet.

3.6.2 Comparaison des visualisations sur CIFAR10

La figure 3.17 présente les visualisations calculées pour 5 exemples du jeu de données cifar10 [87] contenant des images en couleur de dimensions $32 \times 32 \times 3$ et 10 classes. Les visualisations obtenues via la méthode gradCAM et celles via CNN eyes visions sont très similaires. GradCAM semble fournir des visualisations plus floues.

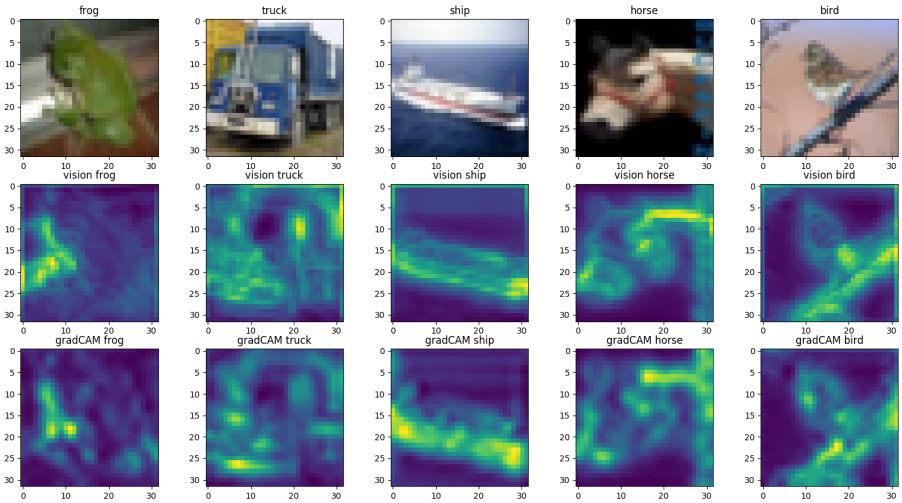


FIGURE 3.17: Comparaison des visualisations obtenues avec GradCAM et CNN eyes visions sur le jeu de données cifar 10 [87]

3.6.3 Comparaison des visualisations sur des IRM 3D

Sur un jeu de données d'imagerie médicale contenant des IRM 3D du cerveau, les visualisations obtenues sont une nouvelle fois similaires via les méthodes gradCAM et CNN eyes visions puisque les mêmes zones du cerveau sont mises en évidence. Cependant gradCAM produit des visualisations avec l'extérieur du cerveau plus intense.

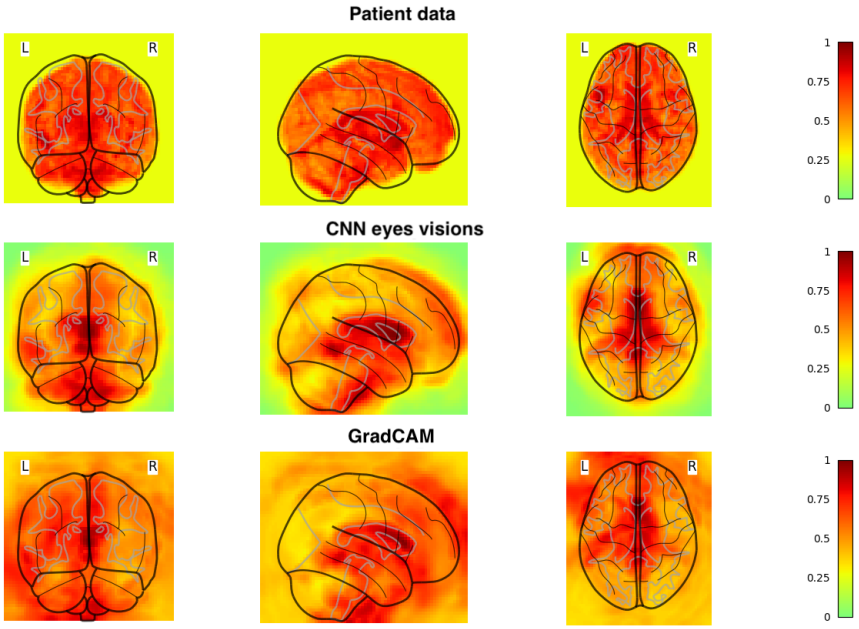


FIGURE 3.18: Comparaison des visualisations obtenues avec GradCAM et CNN eyes visions sur un jeu de données IRM

3.7 Logiciel de visualisation 3D

Dans le but d'utiliser la méthode de visualisation des zones discriminantes CNN eyes visions, un logiciel a été développé durant ces travaux de thèse.
Il permet de charger un modèle ainsi qu'un jeu de données 3-dimensions et d'afficher les résultats directement dans le logiciel. La figure 3.19 présente l'interface du logiciel, avec en haut à gauche le choix du modèle et du jeu de données, au milieu l'affichage des résultats de visualisation et à droite les paramètres d'affichage et de sauvegarde.
Une fois le modèle chargé, l'utilisateur a la possibilité de sélectionner les couches du modèle qu'il souhaite visualiser et le fait de sélectionner plusieurs couches fusionne directement les résultats de chaque couche.
Il en est de même pour le jeu de données, qui une fois chargé laisse la possibilité à l'utilisateur

de sélectionner les données à visualiser et le fait d'en sélectionner plusieurs permet d'obtenir la visualisation moyenne sur l'ensemble des données sélectionnées.

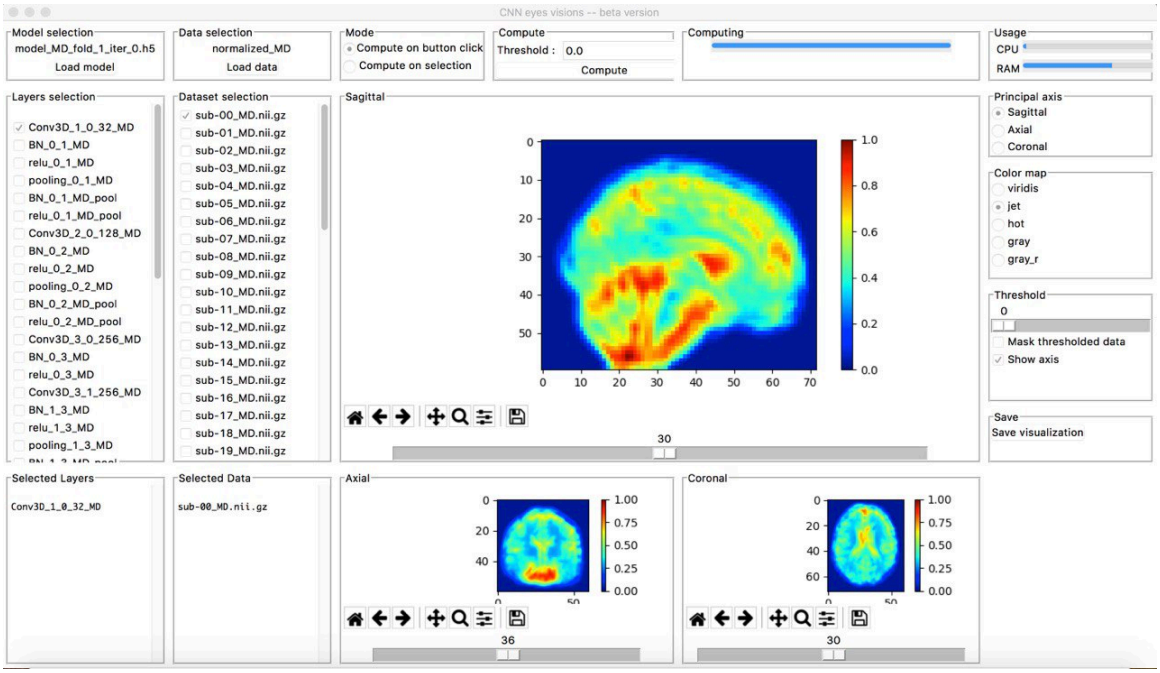


FIGURE 3.19: Logiciel de visualisation des zones discriminantes d'une image 3D

Une amélioration est envisageable sur ce logiciel avec l'ajout de toutes les méthodes de visualisation présentées durant ce chapitre. Une checkbox de séléction permettrait à l'utilisateur de choisir la méthode à employer.

3.8 Conclusion sur les méthodes de visualisation

Ce chapitre passe en revue les différentes méthodes de visualisation des zones discriminantes pour un CNN.
La méthode d'occlusion partielle de l'entrée est simple à comprendre et à mettre en place, cependant l'utilisateur doit définir la taille du filtre d'occlusion. De plus le temps de calcul est très élevé pour obtenir une visualisation.
Les saliency map utilisent les problèmes inverses pour retrouver les pixels pour lesquels une petite variation implique une grosse variation sur les prédictions. La méthode est alors relativement simple à mettre en place mais les temps de calcul sont assez longs.
CAM et gradCAM sont les méthodes les plus utilisées dans l'état de l'art. CAM est simple à mettre en place mais nécessite une architecture particulière pour pouvoir être utilisée alors que gradCAM est plus complexe mais s'applique sur la pluspart des architectures.
CNN eyes visions peut s'appliquer sur tout type d'architecture et les temps de calcul sont très faibles.

La table 3.2 rassemble la totalité de ces informations en prenant en compte la simplicité, le temps de calcul et la précision de la visualisation obtenue. La simplicité correspond à la facilité pour

comprendre la méthode ainsi que le développement du code permettant d'obtenir une visualisation. La précision quant à elle permet d'évaluer à quel point la visualisation des zones discriminantes calculée ressemble aux zones réellement discriminantes. Ces deux termes pouvant être subjectif la table 3.2 utilise les signes + et − pour évaluer ces critères.

Méthode	Simplicité	Temps de calcul	Précision
Occlusion	++	$\sim 10min$	−
Saliency map	++	~ 2	+
CAM	++	$\sim 1sec$	++
GradCAM	−	$\sim 2sec$	++
CNN eyes visions	++	$\sim 0.2sec$	++

TABLE 3.2: Bilan sur les différentes méthodes de visualisation des zones discriminantes d'un CNN

Dans le cas de la neuro imagerie 3-dimensions, et plus particulièrement sur les applications cliniques où le nombre de données est limitées, il est fréquent d'utiliser une validation croisée réitérée. Dans le cas d'une validation croisée à 10 plis réitérée 10 fois, il y a 100 modèles a interpréter, avec une visualisation à calculer par donnée et ce pour chacun des 100 modèles. La méthode CNN eyes visions prend alors tout son sens avec son temps de calcul très court.

CHAPITRE 4

Validation sur des données simulées

4.1 Créations d'un jeu de données simulées

4.1.1 Objectifs des images simulées

L'objectif de la création d'images médicales simulées est d'obtenir un jeu de données avec une vérité terrain disponible, que ce soit pour la classe ou pour la position de la zone discriminante. Deux zones du cerveau ont plus particulièrement été étudiées durant ces travaux de thèse, le cervelet et le putamen. Les figures 4.1 et 4.2 montrent leur position dans le cerveau.

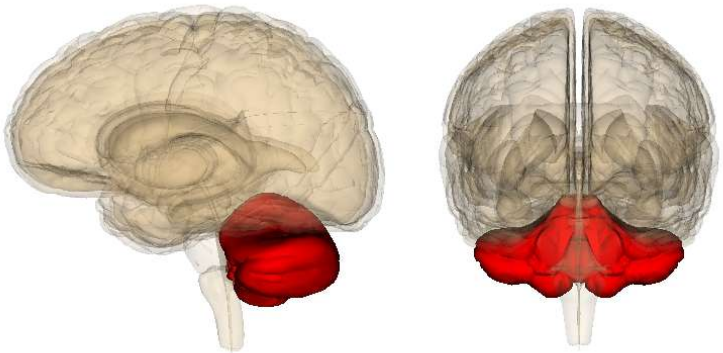


FIGURE 4.1: Position du cervelet dans le cerveau

Afin de créer un jeu de données contenant ces images simulées, la base d'un jeu de données de sujets sains de différents âges a été utilisée. Ainsi la première classe est composée de ces sujets sains. La deuxième classe est ensuite créée en isolant la région d'intérêt (cervelet ou putamen) pour y

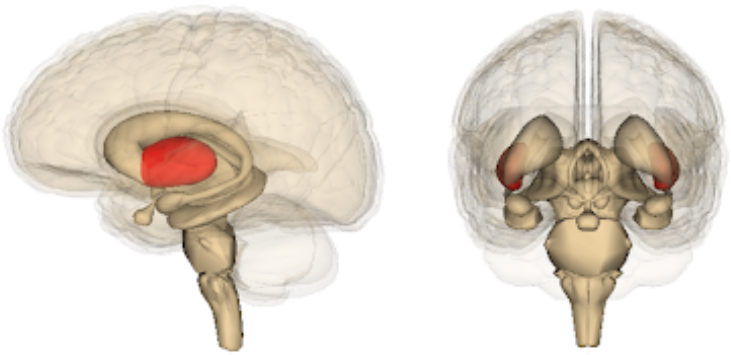


FIGURE 4.2: Position du putamen dans le cerveau

appliquer une modification, et ce pour chaque sujet sain. Le jeu de données est donc finalement composé de 89 sujets sains et 89 patients avec une anormalité induite.

Plusieurs versions de simulation "naïve" sont ensuite présentées dans la section 4.1.2. Puis la version retenue dans les futurs travaux est détaillée dans la section 4.1.3 et correspond aux travaux de thèse de Giulia Maria Mattia, doctorante à l'Inserm Toulouse NeuroImaging Center avec qui j'ai collaboré.

4.1.2 Augmentation de l'intensité

Les premières versions de la création de ce jeu de données sont basées sur l'augmentation de l'intensité de la région d'intérêt (ROI). La figure 4.3 montre un exemple d'une augmentation constante de l'intensité pour chaque point de la ROI. L'équation 4.1 détaille le calcul effectué avec *image* l'image du sujet sain, *ROI* les indices de la région d'intérêt et x la valeur d'augmentation de l'intensité.

$$image[ROI] = image[ROI] + x \tag{4.1}$$

La figure 4.4 montre cette fois-ci un exemple d'une augmentation de l'intensité proportionnelle à l'intensité du point courant et ce pour chaque point de la ROI. L'équation 4.2 détaille le calcul effectué avec *image* l'image du sujet sain, *ROI* les indices de la région d'intérêt et x le pourcentage d'augmentation de l'intensité.

$$image[ROI] = image[ROI] \times (1 + x) \tag{4.2}$$

Ces deux méthodes sont basiques et simples, mais permettent d'obtenir des premiers résultats.

4.1.3 Amélioration de l'augmentation de l'intensité

Afin d'améliorer ces travaux sur la simulation d'un jeu de données IRM, une collaboration a été mise en place avec Giulia Maria Mattia, doctorante à l'Inserm Toulouse NeuroImaging Center. En

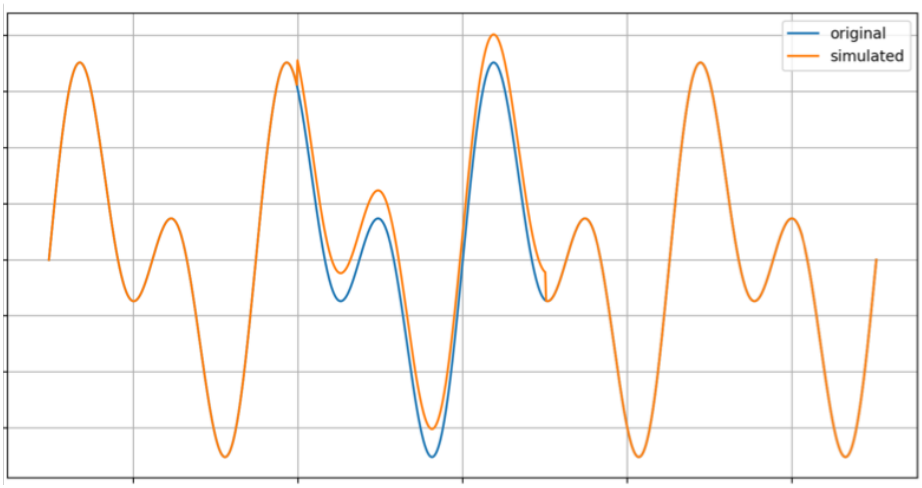


FIGURE 4.3: Simulation d'une altération sur une zone précise d'un signal par augmentation constante de l'intensité

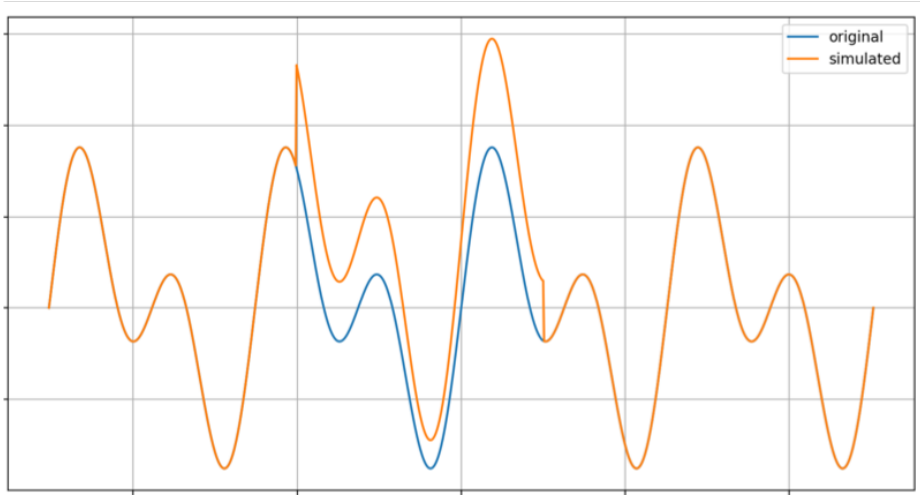


FIGURE 4.4: Simulation d'une altération sur une zone précise d'un signal par augmentation de l'intensité d'un pourcentage du point courant

effet ces travaux utilisent aussi les images médicales simulées et Giulia a produit des images plus proches de la réalité.
Dans ces travaux, les voxels de la ROI ne sont pas tous modifiés, mais seulement les voxels ayant une intensité inférieure à un certain pourcentage de l'intensité maximale de la ROI. Ceci permet d'éviter les problèmes de saturation lorsque l'intensité de modification appliquée est élevée. De plus, en analysant des images réelles de patients atteints de pathologies neurodégénératives, Giulia a pu remarquer que les patients avaient une intensité plus élevée que les sujets sains dans certaines zones du cerveau [88].
La figure 4.5 présente la méthode employée pour simuler une IRM avec une anormalité induite. Plus

d'information sur la méthode de simulation peuvent être trouvée dans [88].

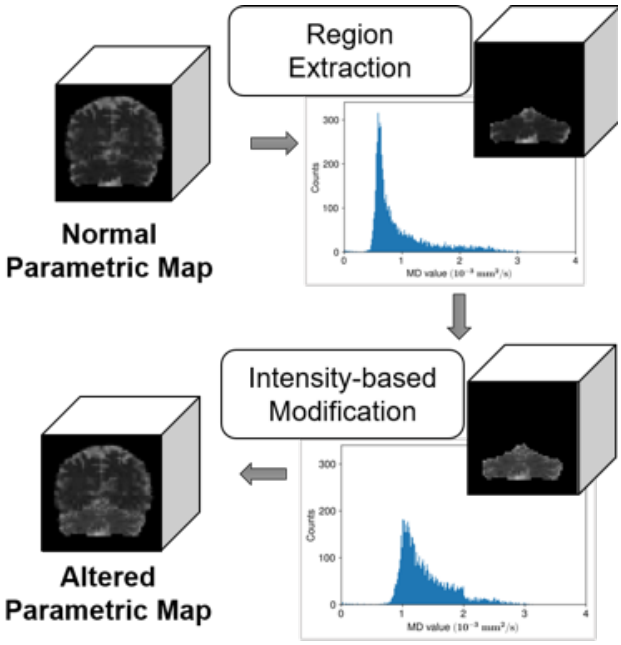


FIGURE 4.5: Simulation d'une altération sur une zone précise d'une IRM via la méthode améliorée [88]

4.2 Amplitude des modifications et effets

4.2.1 Scores des versions de base des images simulées

Les jeux de données présentés dans la section 4.1 ont été utilisés pour alimenter un réseau de neurones convolutifs 3-dimensions basé sur une adaptation du modèle VGG dont le détail est précisé dans la section 5.2.1 et ce pour différentes augmentations d'intensité. La figure 4.6 présente tous les scores d'exactitude obtenus pour les différentes augmentations d'intensité avec la méthode naïve qui consiste à augmenter l'intensité des voxels de la ROI de façon proportionnelle au voxel courant.
Il est alors possible de constater que pour des augmentations d'intensité très faibles, les CNN ne sont pas capables de discriminer les sujets sains des patients avec une anormalité induite.
Le cervelet nécessite une augmentation d'intensité de 20 % pour obtenir un score d'exactitude de 100 %. Le putamen, quant à lui a besoin d'une augmentation de 50 %.
La combinaison des deux régions cervelet et putamen ne nécessite que 15 % d'augmentation d'intensité pour atteindre les 100 % d'exactitude.

Le cervelet étant un organe plus grand que le putamen (environ 7000 voxels pour le cervelet, contre moins de 500 pour le putamen), la figure 4.6 montre que plus grande est la zone dans laquelle une anormalité est induite, moins l'augmentation d'intensité est nécessaire pour qu'un CNN puisse discriminer les sujets sains des patients.

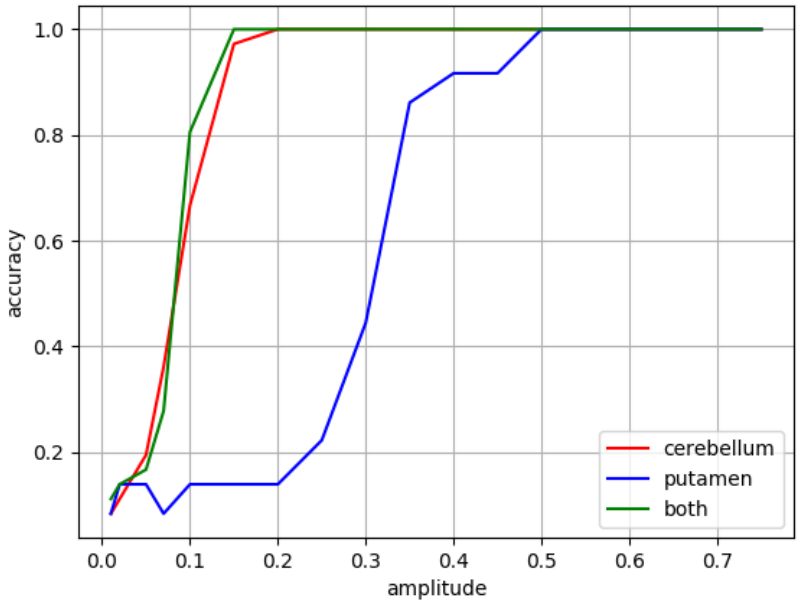


FIGURE 4.6: Score d'exactitude en fonction de l'intensité d'augmentation des images simulées avec la méthode où chaque voxel est augmenté d'un pourcentage du voxel courant

4.2.2 Scores de la version améliorée des images simulées

Les mêmes travaux ont été réalisés sur la version des données simulées créée par Giulia Maria Mattia. Ces travaux permettent d'arriver à la même conclusion comme le montre la figure 4.7. En effet le cervelet n'a besoin d'une augmentation d'intensité que de 27 % pour que le CNN atteigne les 100 % d'exactitude alors que le putamen nécessite 81 %.

Afin de confirmer cette conclusion que plus une zone anormale est grande, moins la modification de l'anormalité nécessite d'être élevée, la version érodée du cervelet et une version dilatée du putamen ont été créées. Le cervelet érodé possède alors une taille similaire au putamen, tandis que le putamen dilaté possède une taille similaire au cervelet. La figure 4.6 montre alors que la courbe d'exactitude en fonction de l'intensité d'augmentation du cervelet érodé (E-cerebellum) est plus proche de celle du putamen. Inversement, la courbe du putamen dilaté (D-putamen) se rapproche de celle du cervelet.

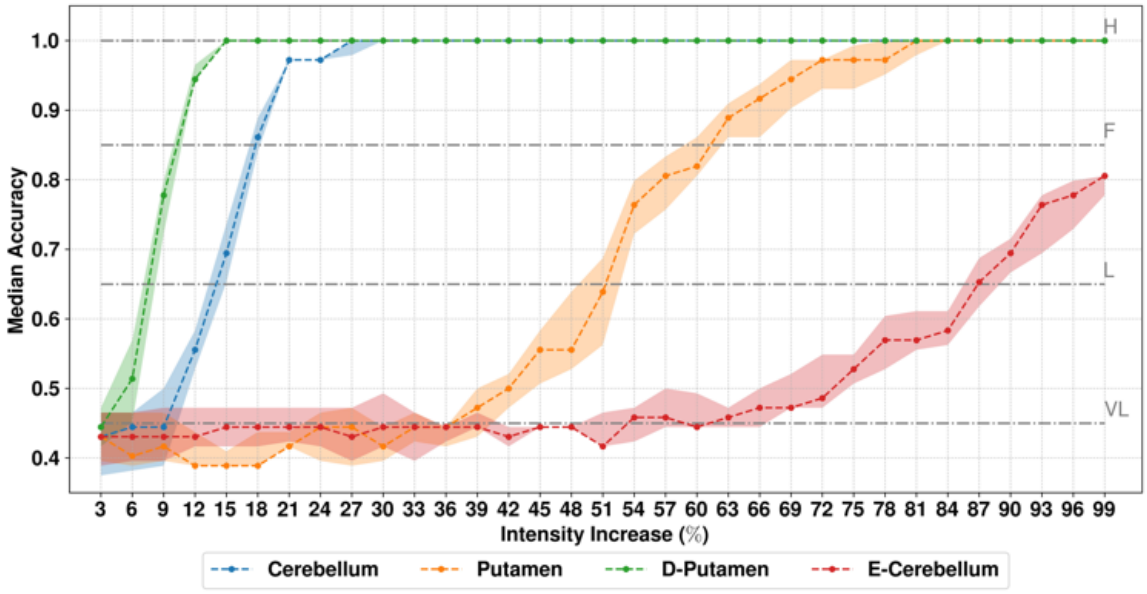


FIGURE 4.7: Score d'exactitude en fonction de l'intensité d'augmentation des images simulées avec la méthode améliorée [88]

4.3 Visualisation d'un CNN sur les données simulées

4.3.1 Visualisation des versions de base des images simulées

La méthode de visualisation CNN eyes visions (voir section 3.5) a été testée sur les données d'IRM 3-dimensions simulées de façon à avoir la position des zones discriminantes disponibles. La section courante présente les résultats de visualisation obtenus sur la version de base des images simulées où chaque voxel voit son intensité augmentée d'un pourcentage de sa valeur.
Les positions des zones où l'anormalité a été induite sont détaillées sur les figures 4.1 et 4.2 de la section 4.1.

Les trois figures suivantes 4.8, 4.9 et 4.10 présentent les résultats obtenus pour le cervelet, le putamen et la combinaison des deux ROI avec une augmentation de l'intensité de 55 %. Autrement dit, cela correspond à une intensité qui permet aux modèles CNN d'obtenir un score d'exactitude de 100 % sur la discrimination entre les sujets sains des patients possédant l'anormalité.
Pour chaque figure, la visualisation moyenne de la classe des sujets sains est en haut des figures, au milieu se trouve la visualisation moyenne de la classe des patients possédant l'anormalité dans la ROI et en bas la différence absolue entre les visualisations moyennes de chaque classe. Le contour vert des figures correspond à la ROI.

La figure 4.8 montre que le cervelet est bien mis en évidence par la méthode de visualisation des zones discriminantes CNN eyes visions.

Tout comme pour le cervelet, la figure 4.9 montre que le putamen est bien mis en évidence par la méthode de visualisation des zones discriminantes CNN eyes visions.

La figure 4.10 quant à elle montre que la visualisation des zones discriminantes pour la com-

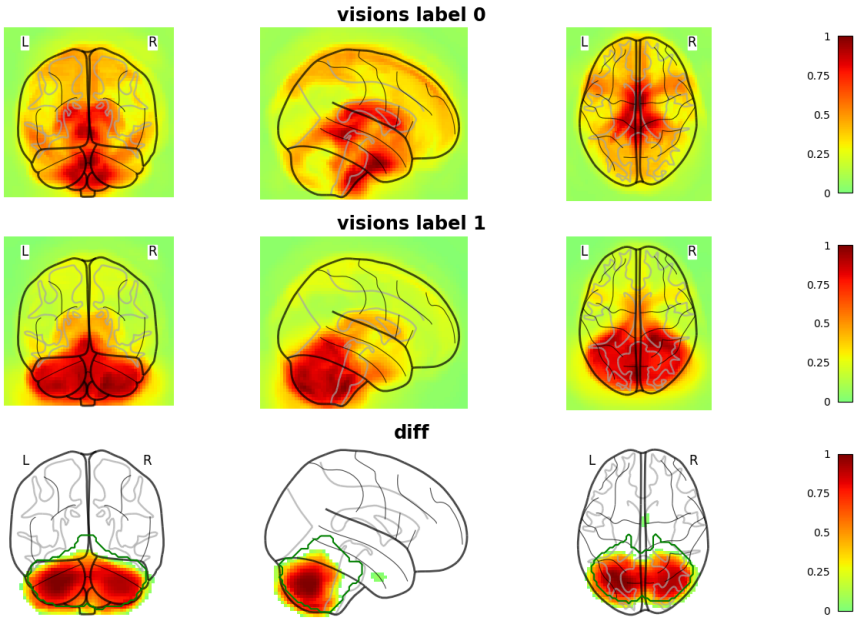


FIGURE 4.8: Visualisation des zones discriminantes des images simulées pour le cervelet avec la méthode où chaque voxel est augmenté d'un pourcentage de 55% du voxel courant

binaison du cervelet et du putamen ne se concentre que sur le cervelet. Une explication est alors envisageable et plausible, puisqu'en effet la taille du putamen très petite devant le cervelet. Le CNN n'aurait alors qu'à s'appuyer sur le cervelet pour discriminer les deux classes.

4.3.2 Visualisation de la version améliorée des images simulées

La visualisation des zones discriminantes via la méthode CNN eyes visions a ensuite été testée sur la version améliorée des données IRM 3-dimensions simulées. Les figures 4.11, 4.12 et 4.13 montrent une nouvelle fois les résultats de visualisation sur le cervelet, le putamen et la combinaison des deux ROI et sont issues d'un article publié en collaboration avec Giulia Maria Mattia [86]. Pour plus d'information, l'article complet est disponible en annexe dans la section 5.5.

Cette fois ci, la méthode de visualisation est appliquée sur un modèle ayant obtenu un score d'exactitude de 100 % et un autre avec un score de 60 % et ce pour les données d'entrainement mais aussi de test. Ceci permet d'éprouver la méthode de visualisation CNN eyes visions à la fois sur un modèle qui discrimine parfaitement les données, et sur un modèle qui ne parvient pas réellement à les discriminer.

La figure 4.11 concernant le cervelet montre que la ROI est bien mise en évidence sur les données d'entrainement et de test pour le modèle ayant un score d'exactitude de 100 %. Concernant le

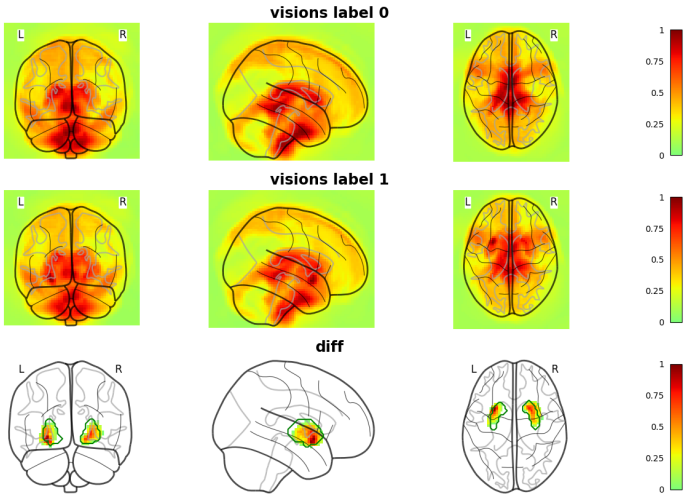


FIGURE 4.9: Visualisation des zones discriminantes des images simulées pour le putamen avec la méthode où chaque voxel est augmenté d'un pourcentage de 55% du voxel courant

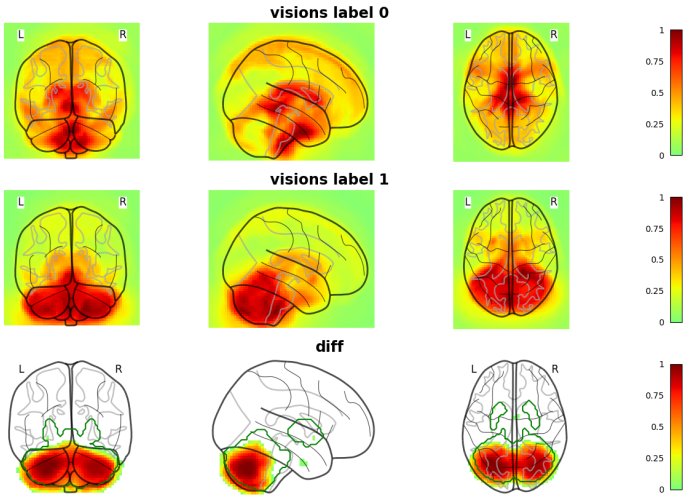


FIGURE 4.10: Visualisation des zones discriminantes des images simulées pour le cervelet et le putamen avec la méthode où chaque voxel est augmenté d'un pourcentage de 55% du voxel courant

modèle ne parvenant pas à discriminer les données, sans surprise la visualisation ne cible pas le cervelet sur les données de test. Cependant, le cervelet est tout de même mis en avant sur les données d'entrainement. Cela permettrait d'obtenir de l'information sur la position de la ROI en utilisant les données d'entrainement, même sur un modèle ayant un score d'exactitude faible.

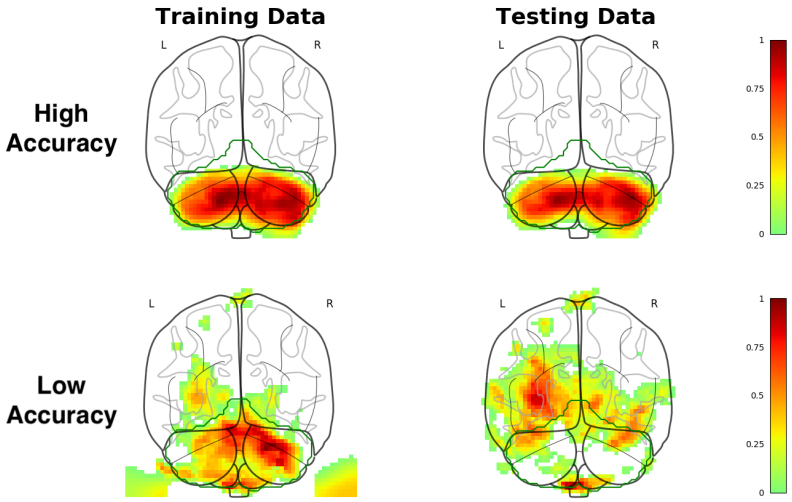


FIGURE 4.11: Visualisation des zones discriminantes des images simulées pour le cervelet pour un modèle ayant une exactitude élevée (100 %) et un modèle ayant une exactitude faible (60 %) [86]

Tout comme pour le cervelet, la figure 4.12 concernant le putamen montre que la ROI est bien mise en évidence sur les données d'entrainement et de test pour le modèle ayant un score d'exactitude de 100 %. Une nouvelle fois le putamen est peu mis en évidence sur les données de test via le modèle ayant un score d'exactitude faible, mais est bien mis en avant sur les données d'entrainement.

Les résultats de visualisation sur la figure 4.13 pour la combinaison du cervelet et du putamen rejoignent les résultats obtenus avec la version de base des images simulées. Le modèle semble privilégier le cervelet au détriment du putamen pour le modèle ayant un score d'exactitude de 100 %. Concernant le modèle avec un faible score d'exactitude, un putamen semble vouloir apparaitre sur les données de test mais il y a présence de diffusion autour du putamen, ce qui laisse difficile l'interprétation des visualisations sur les données de test. Sur les données d'entrainement le cervelet est mis en avant et le putamen est bien actif, mais avec une intensité moindre que le cervelet.

4.3.3 Visualisation appliquée à un sujet unique

La méthode de visualisation CNN eyes visions a été employée pour une application à un sujet unique en comparant la visualisation d'un sujet avec les visualisations moyennes de chaque classe. La figure 4.14 montre alors qu'un sujet de la classe 0 a plus de différence avec la moyenne de la classe 1, et inversement pour le sujet de la classe 1. De plus ces différences sont localisées dans le cervelet pour les deux sujets.
Ainsi la méthode CNN eyes vision permet de déduire visuellement la classification effectuée par le

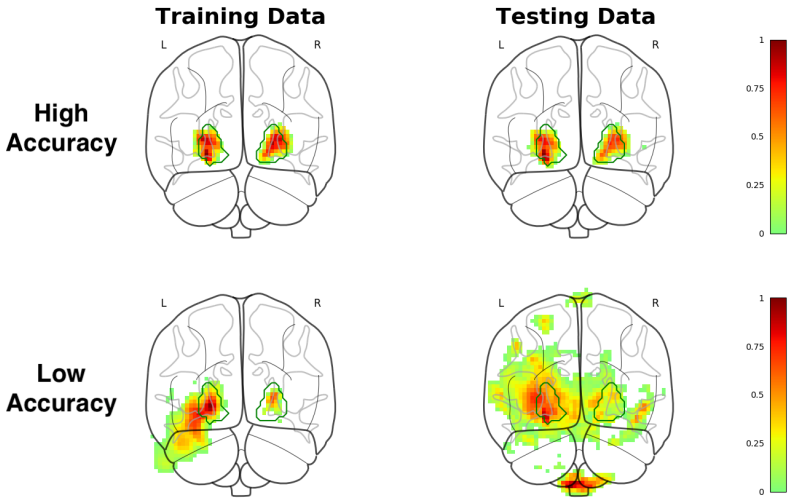


FIGURE 4.12: Visualisation des zones discriminantes des images simulées pour le putamen pour un modèle ayant une exactitude élevée (100 %) et un modèle ayant une exactitude faible (60 %) [86]

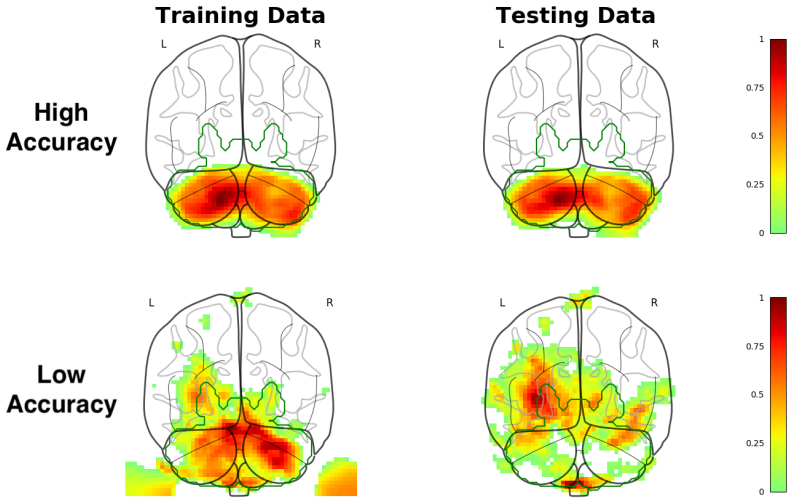


FIGURE 4.13: Visualisation des zones discriminantes des images simulées pour le cervelet et le putamen pour un modèle ayant une exactitude élevée (100 %) et un modèle ayant une exactitude faible (60 %) [86]

modèle CNN, mais aussi les zones sur lesquelles s'est appuyé le modèle pour classer un sujet.

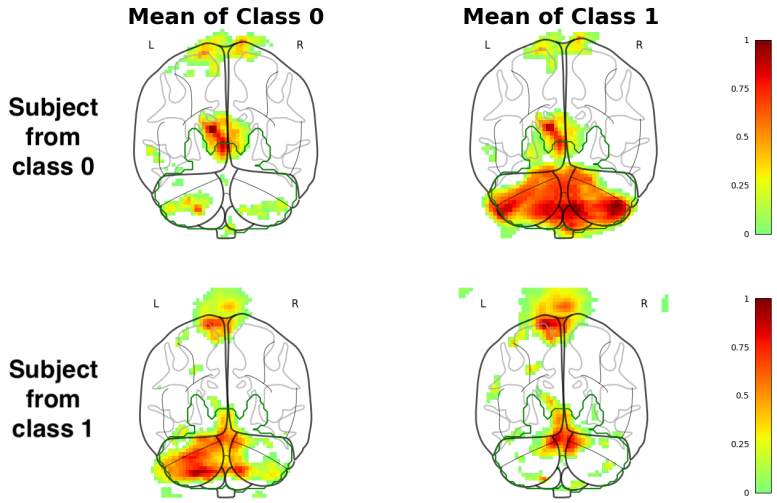


FIGURE 4.14: Visualisation des zones discriminantes des images simulées pour un sujet sain et un patient possédant l'anormalité dans le cervelet et le putamen [86]

CHAPITRE 5

Application aux syndromes Parkinsoniens

5.1 De l'IRM aux prédictions et visualisations

L'intelligence artificielle est utilisée dans de plus en plus de domaines et en particulier l'imagerie médicale. Les modèles dit de "deep learning" fournissent des résultats prometteurs et l'émergence de méthodes d'interprétation de ces modèles considérés longtemps comme des "boites noires" rendent leurs utilisations très intéressantes dans le domaine de l'imagerie médicale.

En combinant les réseaux de neurones convolutifs (CNN) (voir section 2.3), les modèles multi-modaux (voir section 2.8) ainsi que la méthode de visualisation des zones discriminantes CNN eyes visions (voir section 3.5) un pipeline entièrement basé sur le deep learning a été développé durant ces travaux de thèse.
La validation de ces travaux ayant été effectuée sur des données simulées (voir chapitre 4), cela permet d'appliquer ce pipeline sur des données réelles sereinement.

Ce chapitre concerne alors l'application de ce pipeline de deep learning aux syndromes Parkinsoniens et plus particulièrement l'atrophie mutli systématisée (AMS).
Les sous-sections 5.1.1, 5.1.2 et 5.1.3 présentent respectivement l'atrophie multi systématisée, le jeu de données multimodales 3-dimensions utilisé pour la détection de l'AMS ainsi que le benchmark employé.
La section 5.2 détaille les différentes architectures utilisées ainsi que les scores obtenus. La section 5.3 propose une analyse du pouvoir discriminant des différents biomarqueurs et la section 5.4 montre les résultats de visualisation des zones discriminantes pour les architectures employées.

5.1.1 Atrophie Multi Systématisée

L'atrophie multi systématisée est un syndrome Parkinsonien rare dont l'estimation moyenne de l'incidence est de 0.6 cas pour 100 000 personnes-année et une prévalence de 3.4 à 4.9 cas pour 100 000 personnes [56]. L'AMS est caractérisée par l'accumulation de α-synuclein dans les oligodendro-

cytes [89]. Le diagnostic de l'AMS est un challenge et plus particulièrement dans les phases initiales de la pathologie de par le recouvrement entre ses symptômes et ceux des différents syndromes Parkinsoniens.

Plusieurs études proposent une approche basée sur l'intelligence artificielle pour discriminer les principaux syndromes Parkinsoniens, telles que E. Adeli et. al en 2016 [58], F.D. Bowman en 2016 [59], Y. Chen et. al en 2015 [60], D. Long et. al en 2012 [61], B. Peng et. al en 2017 [62] et D. Zhang et. al en 2014 [63]. La plupart ont utilisé des biomarqueurs dérivés de l'IRM tels que le niveau de substance grise issu des images pondérées en T1, l'index d'intégrité microstructural de la matière blanche issu de l'imagerie de diffusion et des biomarqueurs mesurant l'activité et la connectivité du cerveau issus de l'imagerie fonctionnelle. Ces travaux ont atteint des scores de discrimination satisfaisants, de l'ordre de 80 % en moyenne, et ont mis en évidence certaines zones du cerveau considérées comme discriminantes.

D'autres travaux ont utilisé l'intelligence artificielle et les images IRM pour discriminer les sujets sains et les syndromes Parkinsoniens entre eux, l'AMS et la paralysie supranucléaire progressive (PSP) tels que G. Barbagallo et. al en 2016 [57], N.K. Focke et. al en 2011 [31], P. Péran et. al en 2018 [47], H.J Huppertz et. al en 2016 [64] et C. Scherfler et. al en 2016 [90].
Huppertz et al. [64] a utilisé des données volumétriques sur la matière grise et blanche issues d'images pondérées en T1 pour discriminer les patients AMS et les sujets sains ainsi que les patients AMS et les autres syndromes Parkinsoniens avec des scores d'exactitude allant de 60 à 90 %. Le groupe de P. Péran et ses collègues [57, 47] a utilisé des protocoles plus complets en incluant les informations volumétriques et les biomarqueurs dérivés de l'IRM. Chacune de ces études a atteint des scores d'exactitude satisfaisants de l'ordre de 90 %.

Plus récemment, F. Nemmi et. al [46] a développé un pipeline de machine learning multimodal par voxels basé sur les machines à vecteur de support en incluant des biomarqueurs dérivés de l'IRM structurelle, de diffusion et fonctionnelle. Ce pipeline permet de discriminer les sujets sains des patients AMS et des patients Parkinson avec des scores d'exactitudes allant de 78 à 94 % mais aussi d'extraire les zones discriminantes de chacun des biomarqueurs.

Plusieurs autres travaux ont déjà utilisé avec succès des méthodes basées sur les réseaux de neurones pour discriminer les syndromes Parkinsoniens. Esmaeilzadeh et. al en 2018 ont proposé un CNN 3-dimensions discriminant les syndromes Parkinsoniens via les IRM [32]. Shinde et. al ainsi que Kiryu et. al en Kiryu2019 en 2019 utilisent eux aussi les CNN mais cette fois-ci via des coupes d'IRM 2D [44].
Cependant aucun de ces travaux n'utilise un protocole IRM multimodal complet, incluant de l'imagerie structurelle, de diffusion et fonctionnelle. De plus, ces travaux sont basés sur des jeux de données de grande dimension (de 200 à 600 patients) et appliquent une augmentation de données.

5.1.2 Jeu de données 3-dimensions multimodales

Le jeu de données utilisé est composé de 29 patients AMS recrutés dont chacun possède un diagnostic en accord avec les critères de diagnostic internationaux. Chaque patient possède un score de Hoehn et Yahr inférieur à 4 [91, 92]. De plus, tous les patients ont un historique neurologique et psychologique négatif autre que l'AMS, n'ont ni de tumeur cérébrale, ni de lésion vasculaire cérébrale. Une cohorte de sujets sains complète le jeu de données avec 26 sujets droitiers où l'âge, le sexe et le niveau d'éducation sont en accord avec les caractéristiques des patients AMS.

Chacun des patients et sujets du jeu de données a passé un examen d'imagerie médicale sur une IRM 3 Tesla au Toulouse NeuroImaging Center (ToNIC), laboratoire Inserm UMR 1214 à Toulouse. Les images structurelles, de diffusion et fonctionnelles ont été acquises. Les images structurelles ont ensuite été segmentées en niveau de substance grise , niveau de matière blanche et en fluide cérébrospinal puis normalisées en utilisant CAT12 [93]. Les images de diffusion ont été traitées via le pipeline standard FSL [94] permettant d'obtenir la fraction d'anisotropie et la diffusivité moyenne. Pour finir, les images fonctionnelles ont été traitées avec conn [25] ce qui permet de calculer la fraction de l'amplitude de fluctuation des basses fréquences mesurant l'activité du cerveau au repos [95]. Suite à ces traitements, chaque biomarqueur dérivé de l'IRM a une dimension de $60 \times 72 \times 60$ voxels de $3 \times 3 \times 3mm^3$.

Trois biomarqueurs ont particulièrement été étudiés durant ces travaux de thèse permettant d'allier l'imagerie structurelle, de diffusion et fonctionnelle.

— Le niveau de substance grise (gm)
— La diffusivité moyenne (MD)
— L'amplitude de fluctuation des basses fréquences (ALFF)

Le niveau de substance grise est un biomarqueur structurel, il permet donc d'imager la structure du cerveau. Ce biomarqueur peut être pertinent pour la discrimination des pathologies neurodégénératives car ces patients ont souvent un niveau de substance grise qui a tendance à s'affaisser ou se réduire. La figure 5.1 montre un exemple de niveau de substance grise d'un patient AMS contenu dans le jeu de données.

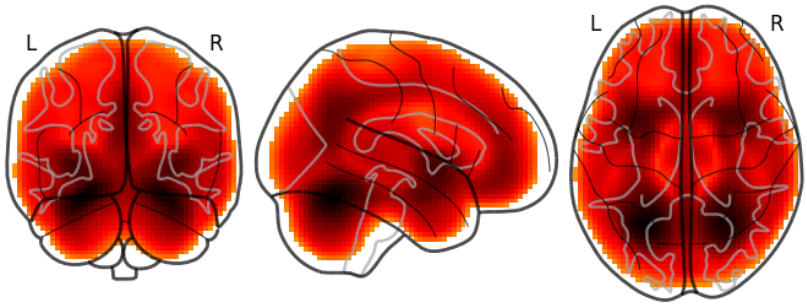


FIGURE 5.1: Biomarqueur dérivé de l'IRM : niveau de substance grise (gm)

La diffusivité moyenne (MD) est un biomarqueur de diffusion qui permet d'imager le mouvement brownien des molécules d'eau. La présence d'agrégats ferreux chez les patients atteints de pathologies neurodégénératives a pour effet de modifier la diffusivité moyenne ce qui fait de MD un bon candidat pour un biomarqueur discriminant. La figure 5.2 montre un exemple de diffusivité moyenne d'un patient AMS contenu dans le jeu de données.

L'amplitude de fluctuation des basses fréquences est un biomarqueur fonctionnel qui permet d'imager l'activité cérébrale au repos. Pour cela, le niveau d'oxygénation du sang dans le cerveau est mesuré dans une séquence IRM temporelle. Ensuite les hautes fréquences sont filtrées de façon à retirer les variations de l'oxygénation du sang dues aux battements du coeur et l'amplitude de fluctuation est conservée de façon à obtenir une image 3-dimensions plutôt qu'une séquence temporelle. La figure 5.3 montre un exemple d'amplitude de fluctuation des basses fréquences d'un patient AMS contenu dans le jeu de données.

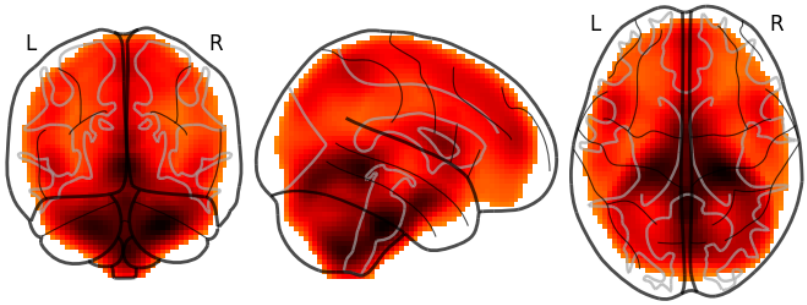


FIGURE 5.2: Biomarqueur dérivé de l'IRM : diffusivitié moyenne (MD)

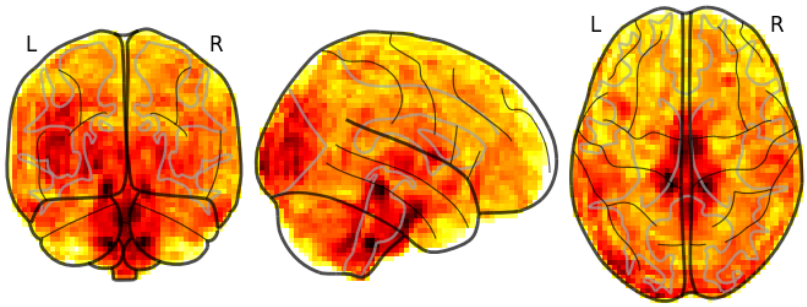


FIGURE 5.3: Biomarqueur dérivé de l'IRM : amplitude de fluctuation des basses fréquences (ALFF)

5.1.3 Benchmark

Pour réaliser l'étude sur l'utilisation de l'intelligence artificielle appliquée à l'atrophie multi systématisée, un benchmark a été mis en place. Celui-ci a été développé de façon à être le miroir du pipeline proposé par F. Nemmi et. al [46], utilisant le même jeu de données. La figure 5.4 montre la comparaison entre le pipeline original proposé par F. Nemmi et celui développé dans ces travaux. Dans les deux cas, le pipeline consiste en :

— L'extraction de caractéristiques,
— La réduction du nombre de caractéristiques,
— La fusion des biomarqueurs,
— La classification

Afin de comparer les résultats obtenus avec le pipeline de deep learning, basé sur les CNN multimodaux 3-dimensions, la même validation croisée à 10 plis réitérée 10 fois que F. Nemmi et. al a été employée.
L'algorithme 3 détaille les différentes étapes de l'expérience. Il permet de tester les différentes architectures de CNN disponibles dans la littérature, pour chaque combinaison des biomarqueurs et de calculer une visualisation des voxels discriminants.

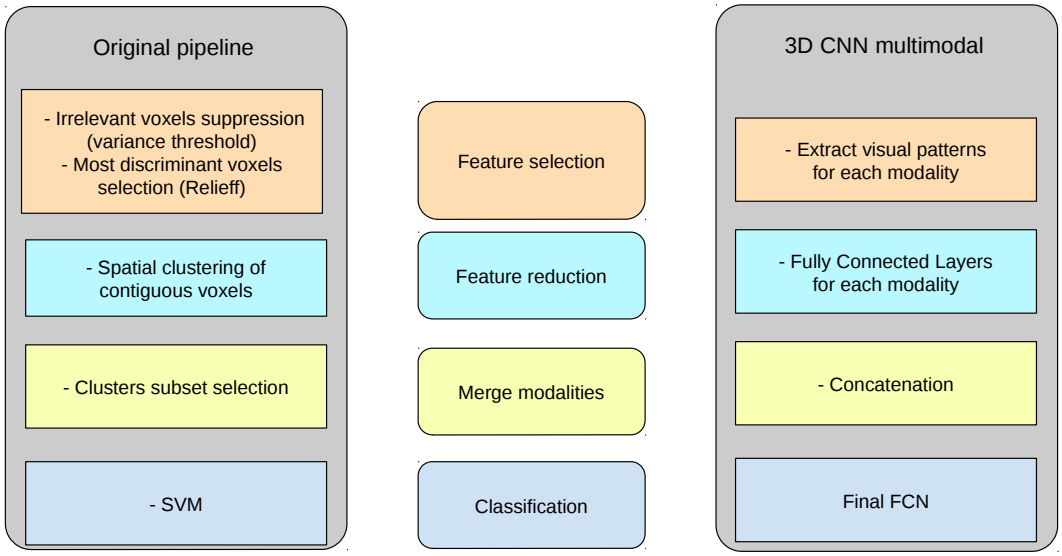


FIGURE 5.4: Comparaison entre le pipeline 3D CNN multimodal (droite) et le pipeline SVM de F. Nemmi et. al [46] (gauche)

Algorithm 3 Calcul des prédictions et visualisations

```
for chaque architecture du modèle do
    for chaque combinaison des biomarqueurs dérivés de l'IRM do
        for chaque itération de la validation croisée do
            for chaque découpe de la validation croisée do
                Pré traitement des données
                Créer le modèle
                entrainer le modèle sur les données d'entrainement
                tester le modèle sur les données de test
                sauvegarde du modèle et des scores du test
                visualisation des zones discriminantes
            end for
            scores sur la totalité du jeu de données
        end for
        moyenne des scores de la validation croisée
        visualisation moyenne des zones discriminantes
    end for
    score de l'architecture
end for
```

5.2 Scores de prédictions

Cette section présente les scores de prédictions obtenus lors de l'adaptation 3-dimensions multimodale des trois architectures de CNN les plus courantes dans la littérature. Chaque section détaille l'architecture du modèle en mono et multimodal puis les scores d'exactitude, de sensitivité et spécificité obtenus.

Afin de pouvoir comparer les résultats, chaque architecture possède le même nombre de couches entièrement connectées et la fusion des modalités est appliquée au même endroit.
Tous les modèles sont initialisés avec la même méthode [70] et la même graine aléatoire (de façon à obtenir des initialisations similaires). De plus, ils sont entrainés avec le même pas d'apprentissage de 5×10^{-5} et le même optimiseur Adam [72, 73] pour une durée de 75 epochs. Tous les modèles emploient aussi la même fonction d'erreur d'entropie croisée (voir section 2.7.6).

De plus, chacun des modèles possède des similarités dans la conception de l'architecture. Les couches de convolution (figure 5.5.a) et celles entièrement connectées (figure 5.5.b) sont suivies d'une couche de batch normalisation permettant de réduire le sur-apprentissage [96] et d'une couche d'activation ELU [66] permettant d'accélérer l'apprentissage. Seule la dernière couche de prédiction (figure 5.5.c) contient un neurone par classe, soit deux neurones dans cette étude, suivie d'une couche d'activation Softmax afin de transformer la sortie en probabilité d'appartenance aux deux classes. Les deux premières couches entièrement connectées contiennent un dropout afin d'éviter le sur-apprentissage [69].

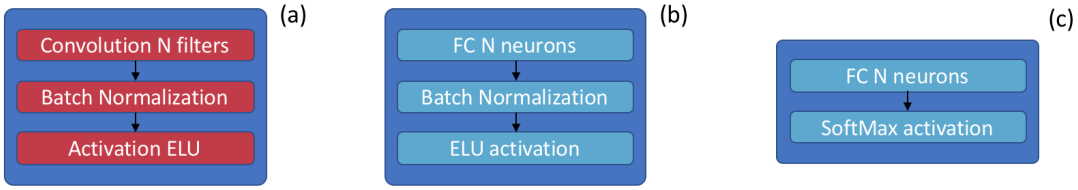


FIGURE 5.5: Détail des couches employées dans la conception des différentes architectures

Les modèles multimodaux emploient tous le même type de fusion tardive en concaténant les sorties des architectures des versions monomodales. Une couche entièrement connectée de $2 \times N$ neurones est insérée avant la prédiction finale du modèle utilisant N modalités comme le montre la figure 5.6.

5.2.1 Adaptation 3-dimensions du modèle VGG

L'architecture basée sur le modèle VGG est composée d'une séquence de couches de convolution et de pooling et se termine par des couches entièrement connectées pour calculer la prédiction [18]. Cette architecture est détaillée précédemment dans la section 2.3.1.

La figure 5.7 détaille l'architecture monomodale développée avec le nombre de couches, le nombre de filtres de convolution ainsi que leur taille.

La table 5.1 présente les scores d'exactitude, de sensitivité et de spécificité obtenus pour chaque combinaison des trois biomarqueurs dérivés de l'IRM. L'exactitude est un score général sur l'ensemble du jeu de données alors que la sensitivité et la spécificité représentent la capacité à correctement classer les patients atteints de pathologies neurodégénératives et les sujets sains, respectivement (voir section 2.6.2).

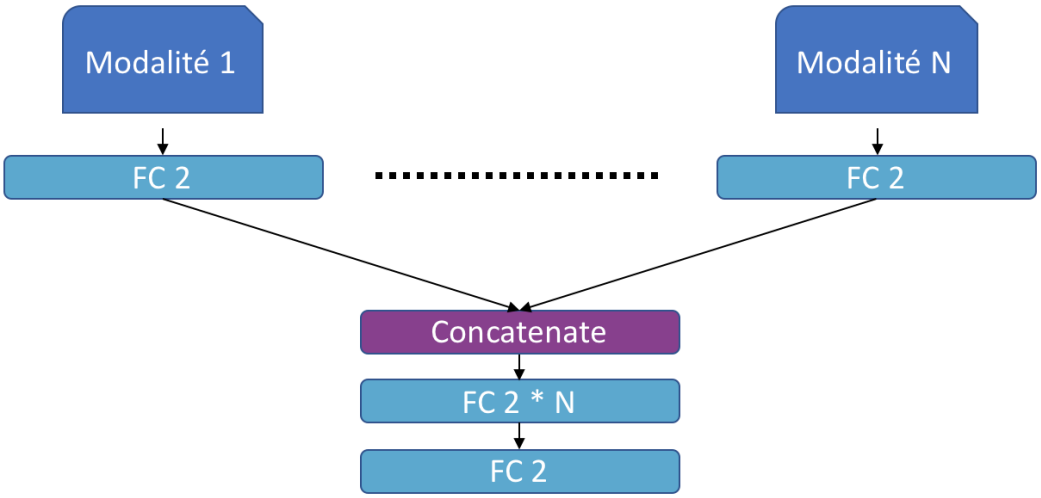


FIGURE 5.6: Détail de la fusion tardive des modèles multimodaux

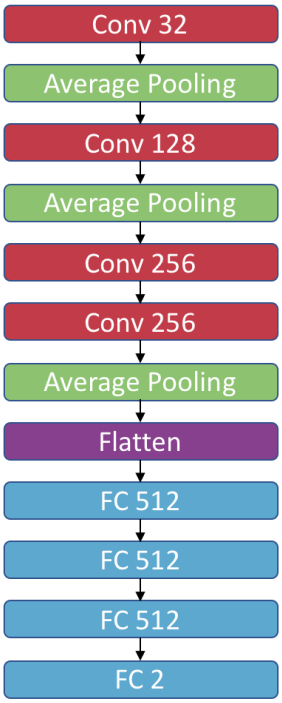


FIGURE 5.7: Détail de l'architecture de l'adaptation 3-dimensions du modèle VGG

	EXACTITUDE	SENSITIVITÉ	SPÉCIFICITÉ
gm	$79.3 \pm 1.7\%$	$72.0 \pm 2.4\%$	$87.3 \pm 1.8\%$
MD	$89.5 \pm 1.8\%$	$87.9 \pm 1.7\%$	$91.2 \pm 2.5\%$
ALFF	$73.1 \pm 3.9\%$	$70.7 \pm 3.9\%$	$75.8 \pm 5.5\%$
gm + MD	$89.1 \pm 2.3\%$	$86.2 \pm 4.4\%$	$92.3 \pm 3.4\%$
gm + ALFF	$84.4 \pm 2.0\%$	$83.8 \pm 2.7\%$	$85.0 \pm 3.6\%$
MD + ALFF	$89.5 \pm 1.6\%$	$91.4 \pm 1.7\%$	$87.3 \pm 3.0\%$
gm + MD + ALFF	$89.1 \pm 2.0\%$	$87.2 \pm 3.5\%$	$91.2 \pm 1.8\%$

TABLE 5.1: Exactitude, sensitivité et spécificité obtenues pour chaque combinaison des différentes modalités avec l'adaptation 3-dimensions du modèle VGG

Le biomarqueur MD obtient donc les meilleurs résultats en monomodal avec une exactitude de $89.5 \pm 1.8\%$ alors que le biomarqueur ALFF produit les moins bons scores de cette étude avec $73.1 \pm 3.9\%$. Tous les modèles multimodaux obtiennent des scores d'exactitude supérieurs à 84%. La combinaison des biomarqueurs MD et ALFF obtient la meilleure sensitivité avec un score de $91.4 \pm 1.7\%$ et permet de mieux reconnaître les patients AMS présents dans le jeu de données.
Les scores de sensitivité et spécificité sont relativement proches hormis pour le biomarqueur GM où l'écart entre ces deux scores est d'environ 15%. Il est donc moins performant pour détecter les patients AMS que les patients sains.

5.2.2 Adaptation 3-dimensions du modèle ResNet

L'adaptation 3-dimensions du modèle ResNet utilise les "skip connections" [19]. Afin de pouvoir appliquer la couche "ADD" qui calcule l'addition de la sortie de plusieurs couches, du "zero padding" est appliqué afin de combler les zones non présentes pour obtenir une dimension souhaitée, dans le but de pouvoir calculer cette addition. La dimension des données est donc réduite plus lentement lors du transit à travers le modèle, ce qui implique un plus grand nombre de couches en comparaison avec le modèle basé sur l'architecture VGG. La figure 5.8 détaille l'architecture du modèle monomodal.

La table 5.2 présente les scores d'exactitude, de sensitivité et de spécificité obtenus pour chaque combinaison des trois biomarqueurs dérivés de l'IRM.

	EXACTITUDE	SENSITIVITÉ	SPÉCIFICITÉ
gm	$76.7 \pm 2.7\%$	$65.5 \pm 4.6\%$	$89.2 \pm 2.3\%$
MD	$85.1 \pm 1.9\%$	$80.0 \pm 3.4\%$	$90.8 \pm 2.6\%$
ALFF	$73.1 \pm 4.8\%$	$65.9 \pm 7.1\%$	$81.2 \pm 2.7\%$
gm + MD	$86.1 \pm 2.5\%$	$84.8 \pm 3.2\%$	$89.2 \pm 2.3\%$
gm + ALFF	$75.5 \pm 4.3\%$	$70.3 \pm 5.4\%$	$81.2 \pm 5.6\%$
MD + ALFF	$74.9 \pm 3.9\%$	$71.4 \pm 5.6\%$	$78.8 \pm 3.5\%$
gm + MD + ALFF	$83.1 \pm 3.0\%$	$74.1 \pm 3.9\%$	$93.1 \pm 5.4\%$

TABLE 5.2: Exactitude, sensitivité et spécificité obtenues pour chaque combinaison des différentes modalités avec l'adaptation 3-dimensions du modèle resNet

Tout comme les modèles basés sur l'architecture VGG, le biomarqueur MD obtient les meilleurs scores en utilisant les modèles resNet avec une exactitude de $85.1 \pm 1.9\%$ et un accord stable entre la sensitivité et la spécificité.

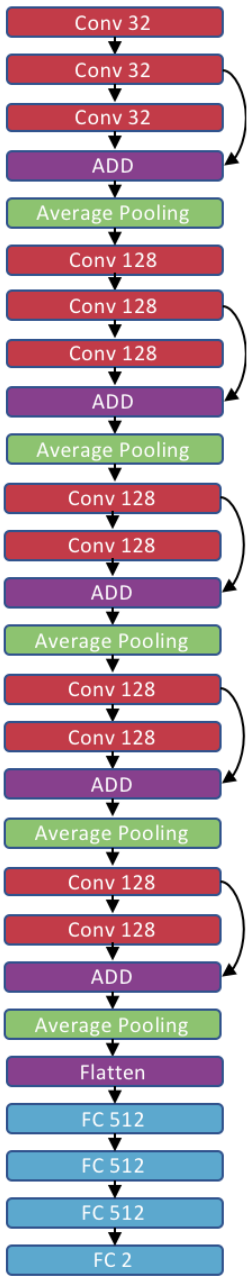


FIGURE 5.8: Détail de l'architecture de l'adaptation 3-dimensions du modèle ResNet

Cependant contrairement aux modèles basés sur l'architecture VGG, ceux basés sur resNet supportent moins bien le multimodal. Les combinaisons des biomarqueurs GM et MD ainsi que GM,

MD et ALFF obtiennent une exactitude supérieure à 83%. La présence d'ALFF dans la combinaison des trois biomarqueurs fait baisser la sensitivité et donc la capacité à reconnaître les patients AMS.

5.2.3 Adaptation 3-dimensions du modèle GoogleNet

L'adaptation 3-dimensions du modèle GoogleNet utilise les modules d'inception [20]. Afin de pouvoir appliquer la couche de concaténation, du "zero padding" " est appliqué comme précdemment, dans le but de pouvoir calculer cette concaténation (voir la figure 5.9). La dimension des données est donc réduite plus lentement lors du transit à travers le modèle, ce qui implique un plus grand nombre de couches en comparaison avec le modèle basé sur l'architecture VGG. La figure 5.10 détaille l'architecture du modèle monomodal.

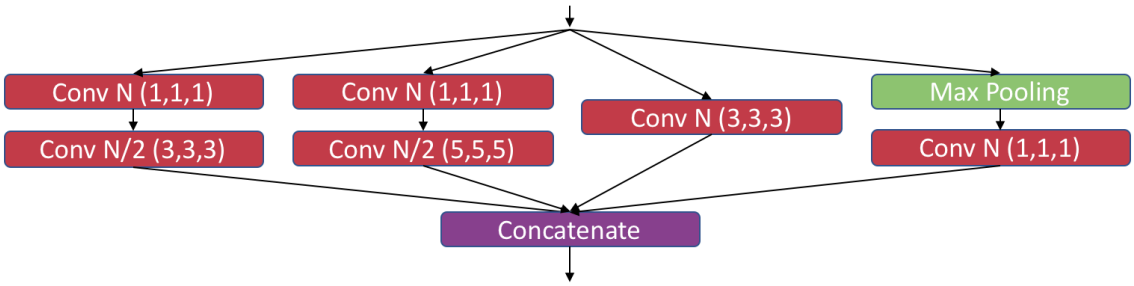


FIGURE 5.9: Détail de l'architecture du module d'inception 3-dimensions

La table 5.3 présente les scores d'exactitude, de sensitivité et de spécificité obtenus pour chaque combinaison des trois biomarqueurs dérivés de l'IRM.

	EXACTITUDE	SENSITIVITÉ	SPÉCIFICITÉ
gm	$80.5 \pm 1.8\%$	$69.0 \pm 3.4\%$	$93.5 \pm 3.7\%$
MD	$88.5 \pm 3.2\%$	$87.9 \pm 4.4\%$	$89.2 \pm 4.1\%$
ALFF	$77.8 \pm 2.9\%$	$73.4 \pm 4.1\%$	$82.7 \pm 3.1\%$
gm + MD	$79.5 \pm 2.0\%$	$66.9 \pm 2.8\%$	$93.5 \pm 3.5\%$
gm + ALFF	$80.7 \pm 2.6\%$	$69.7 \pm 2.6\%$	$93.1 \pm 4.8\%$
MD + ALFF	$89.5 \pm 2.1\%$	$89.3 \pm 2.9\%$	$89.6 \pm 2.5\%$
gm + MD + ALFF	$87.3 \pm 1.8\%$	$90.3 \pm 3.4\%$	$83.8 \pm 2.3\%$

TABLE 5.3: Exactitude, sensitivité et spécificité obtenues pour chaque combinaison des différentes modalités avec l'adaptation 3-dimensions du modèle GoogleNet

Tout comme les modèles basés sur l'architecture VGG, le biomarqueur MD obtient les meilleurs scores en utilisant les modèles googleNet avec une exactitude de $88.5 \pm 3.2\%$ et un accord stable entre la sensitivité et la spécificité.
Les combinaisons des biomarqueurs MD et ALFF ainsi que GM, MD et ALFF obtiennent une exactitude supérieure à 87% et une bonne sensitivité ce qui permet de correctement déctecter les patients AMS.
Le biomarqueur GM souffre d'une faible sensitivité, lorsqu'il est utilisé seul et ce qui se retranscrit sur les combinaisons GM et MD ainsi que GM et ALFF.

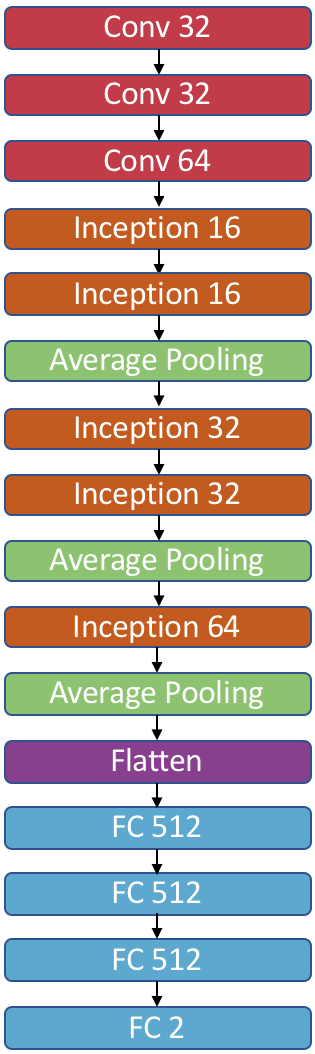


FIGURE 5.10: Détail de l'architecture de l'adaptation 3-dimensions du modèle GoogLeNet

5.2.4 Analyse des différentes architectures

L'analyse des trois tables des scores de prédictions 5.1, 5.2 et 5.3 montre que l'architecture VGG produit les meilleurs résultats en mono et multimodal. De plus, elle supporte le mieux la fusion de modalités car les scores sont améliorés lorsque les deux biomarqueurs les moins discriminants sont fusionnés. Les scores ne sont pas détériorés lorsque le biomarqueur le plus discriminant est utilisé. Aussi, l'architecture VGG permet d'obtenir les meilleurs scores de sensitivité et donc la capacité à détecter les patients AMS, ainsi que le meilleur accord entre sensitivité et spécificité.
L'architecture VGG semble donc la plus indiquée pour discriminer les pathologies neurodégénératives. En effet, les architectures ResNet et GoogLeNet utilisent du "zero-padding" afin de pouvoir appliquer les opérations d'addition et de concaténation. Or l'ajout de ces zéros réduit la propor-

tion d'information utile contenue dans l'image. Cela est d'autant plus important en neuroimagerie 3-dimensions, où l'extérieur du cerveau est composé de zéros et contient donc une grande partie d'information inutile dans l'image d'origine.
De plus, l'architecture originelle de googleNet contient deux sorties de prédiction. Or dans le cas de données 3-dimensions, il est difficile d'employer une deuxième sortie au milieu de l'architecture car la consommation de mémoire devient excessive.

La table 5.4 permet de comparer les trois architectures employées en version mono, bi et tri modale. Plusieurs critères permettent d'analyser ces différentes architectures, à savoir la taille et le nombres de paramètres du modèle, la durée de l'entrainement ainsi que le nombre de filtres de convolution.
Il est alors possible de constater que les architectures de VGG et ResNet ont des tailles similaires, que ce soit en mémoire ou en nombre de paramètres. Ces deux architectures sont plus gourmandes que GoogLeNet. Cependant le modèle VGG est le plus rapide à entrainer et contient moins de filtres de convolution que les deux autres architectures.

		VGG	ResNet	GoogLeNet
Monomodal	Taille	182 Mo	184 Mo	32 Mo
	Paramètres	15 179 586	15 363 090	2 661 314
	Durée de l'entrainement	117 sec	750 sec	324 sec
	Nombre de filtres	672	1 248	928
Bimodal	Taille	364 Mo	369 Mo	64 Mo
	Paramètres	30 359 218	30 738 226	5 322 674
	Durée de l'entrainement	231	1 479 sec	679 sec
	Nombre de filtres	1 344	2 496	1 856
Trimodal	Taille	546 Mo	553 Mo	97 Mo
	Paramètres	45 538 838	46 107 350	7 984 226
	Durée de l'entrainement	373 sec	2 217 sec	1 110 sec
	Nombre de filtres	2 016	3 744	2 784

Table 5.4: Comparaison des architectures

5.3 Pouvoir discriminant des biomarqueurs

L'analyse des pouvoirs discriminants des différents biomarqueurs permet de compléter l'étude et d'obtenir plus d'information sur la pathologie étudiée.
Ces informations sont alors utiles afin de mieux comprendre la pathologie et de déterminer si les modifications sont plutôt dans la structure du cerveau ou dans son fonctionnement. L'analyse du pouvoir discriminant est aussi utile pour la partie imagerie médicale en permettant de mieux cibler l'examen à effectuer pour détecter la pathologie.

En utilisant le pipeline entièrement basé sur le deep learning, cette analyse du pouvoir discriminant des biomarqueurs est effectuée post entrainement en comparant les scores obtenus pour chaque combinaison des biomarqueurs.

L'analyse est donc basée sur les tables de la section 5.2.
Le biomarqueur MD a obtenu les meilleurs scores en monomodal et ce quel que soit le type d'ar-

chitecture employé, ce qui en fait le meilleur candidat pour le biomarqueur discriminant pour la pathologie AMS. De plus en analysant les scores sur les modèles multimodaux, les meilleurs scores sont toujours obtenus pour une combinaison incluant le biomarqueurs MD.

Le biomarqueur ALFF seul obtient les moins bons scores quelle que soit l'architecture utilisée. Ce biomarqueur s'est tout de même montré discriminant avec des scores d'exactitude de l'ordre de 75%. Cependant ALFF ne conserve que l'amplitude de fluctuation des basses fréquences or il existe des architectures de réseaux de neurones capables de traiter les séquences temporelles. Ces modèles basés sur les couches Long Short Term Memory (LSTM) ont été introduites par Hochreiter et. al [97] et ont été adaptés en version convolutive avec les couches conv-LSTM. Ces architectures ont été appliquées avec succès [98] mais aussi dans le domaine médical [99].
Ce biomarqueur pourrait alors être traité par une architecture basée sur les conv-LSTM et permettrait peut-être d'obtenir de meilleurs résultats.

5.4 Visualisation des zones du cerveau incriminées dans les prédictions

Cette section montre les visualisations des voxels discriminants obtenus avec la méthode CNN eyes visions (voir section 3.5), et ce pour chaque biomarqueur et chaque architecture de CNN utilisée à la fois pour les modèles monomodaux et multimodaux.
Trois scores sont apposés sur les figures et correspondent à la ressemblance avec les visualisations obtenues par F. Nemmi et. al [46]. Pour cela, les visualisations obtenues par F. Nemmi et. al ont été considérées comme vérité terrain ce qui permet de calculer des scores d'exactitude, de sensitivité et de spécificité. La sensitivité et la spécificité permettent de quantifier la capacité à retrouver respectivement les voxels actifs et inactifs dans les deux méthodes. Chaque figure possède alors les trois scores entre parenthèses (exactitude, spécificité, sensitivité). Un grand nombre de voxels étant inactifs, en particulier dû à l'extérieur du cerveau, il est normal que l'exactitude et la spécificité soient élevées. Le score le plus intéressant est alors la sensitivité en dernier sur les figures.

5.4.1 Visualisation de l'adaptation 3-dimensions du modèle VGG

La figure 5.11 montre les visualisations des voxels discriminants du biomarqueur GM avec le modèle basé sur l'adaptation 3-dimensions de l'architecture VGG.
Les visualisations des voxels discriminants ont alors mis en évidence le cervelet, les sous régions du tronc cérébrale ainsi que le putamen. Ces régions correspondent à ce que F. Nemmi et. al a mis en évidence [46].
Environ 40 % des voxels actifs sont en communs pour chaque combinaison de biomarqueur utilisant GM. De plus, les visualisations sont similaires quelle que soit la combinaison de biomarqueurs employée.

La figure 5.12 montre les visualisations des voxels discriminants du biomarqueur MD avec le modèle basé sur l'adaptation 3-dimensions de l'architecture VGG.
Les visualisations des voxels discriminants ont alors mis en évidence le cervelet, les sous régions du tronc cérébrale ainsi que le putamen. Ces régions correspondent à ce que F. Nemmi et. al a mis en évidence [46].
Environ 70 % des voxels actifs sont en commun pour chaque combinaison de biomarqueur utilisant MD. De plus, les visualisations sont similaires quelle que soit la combinaison de biomarqueurs employée.

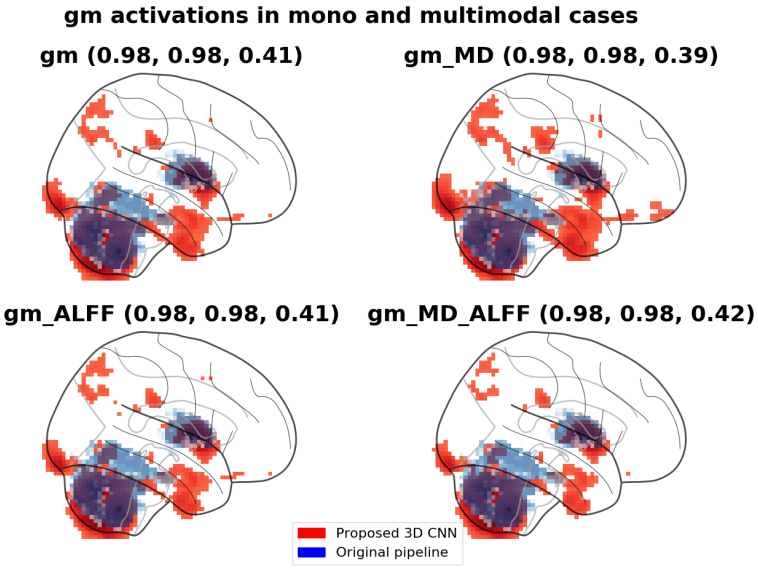


FIGURE 5.11: Visualisation des zones discriminantes avec la méthode CNN eyes visions pour le biomarqueur GM (rouge) et comparaison avec la méthode F. Nemmi et. al [46] (bleu)

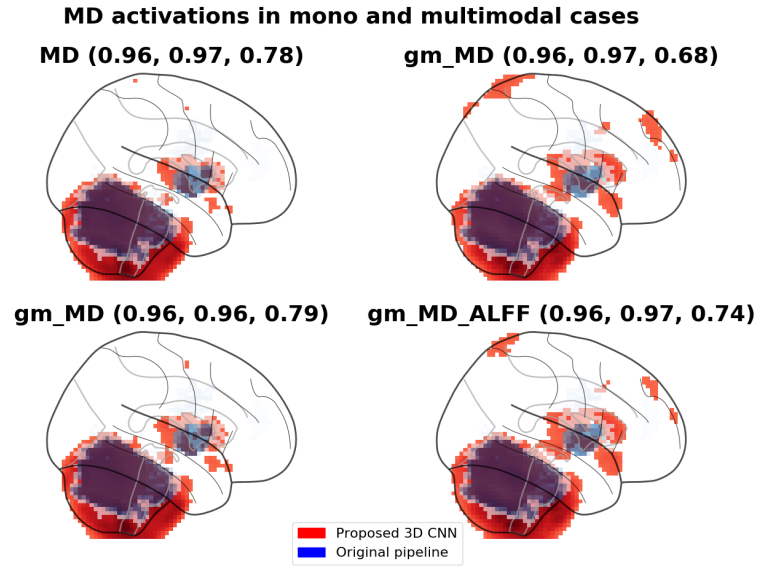


FIGURE 5.12: Visualisation des zones discriminantes avec la méthode CNN eyes visions pour le biomarqueur gm (rouge) et comparaison avec la méthode F. Nemmi et. al [46] (bleu)

La figure 5.13 montre les visualisations des voxels discriminants du biomarqueur ALFF avec le modèle basé sur l'adaptation 3-dimensions de l'architecture VGG.
Les visualisations mettent en évidence des zones du lobe pariétal latéral (en particulier le gyrus angulaire et marginal, bilatéralement) ainsi que des clusters dans le tronc cérébral et le cortex préfrontal dorso-latéral.
Une comparaison avec les visualisations de F. Nemmi et. al n'est pas possible car ce biomarqueur n'a jamais été retenu comme discriminant dans son pipeline.

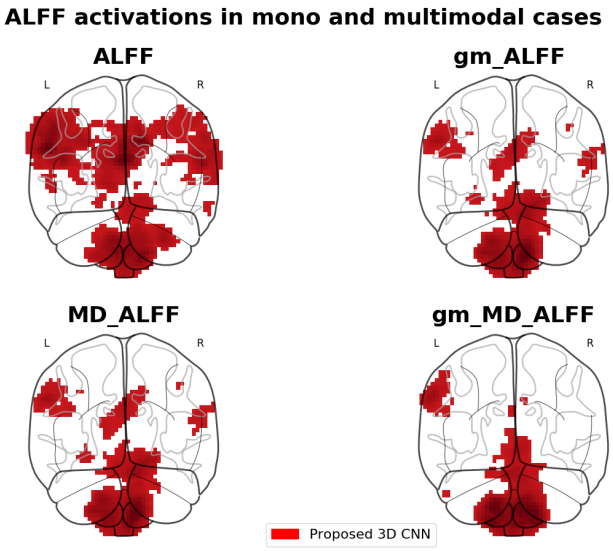


FIGURE 5.13: Visualisation des zones discriminantes avec la méthode CNN eyes visions pour le biomarqueur ALFF

5.4.2 Visualisation de l'adaptation 3-dimensions du modèle ResNet

Les modèles basés sur l'adaptation 3-dimensions du modèle ResNet sont plus profonds que les modèles basés sur VGG. Il y a alors plus de visualisations intermédiaires de très petite dimension ce qui amplifie les effets indésirables de l'interpolation de ces décompositions intermédiaires des couches les plus profondes.

La figure 5.14 montre les visualisations des voxels discriminants du biomarqueur GM avec le modèle basé sur l'adaptation 3-dimensions de l'architecture resNet.
Seule la combinaison des biomarqueurs GM et MD obtiennent des voxels discriminants en commun avec F. Nemmi et. al. Ceci correspond aux scores obtenus (voir table 5.2) puisque seule la combinaison des biomarqueurs GM et MD a une sensitivité élevée et est donc capable de détecter les patients AMS. Autrement dit, seule cette combinaison apprend les caractéristiques correspondant aux patients AMS.
Le cervelet est alors mis en évidence alors que le putamen est omis.

La figure 5.15 montre les visualisations des voxels discriminants du biomarqueur MD avec le modèle basé sur l'adaptation 3-dimensions de l'architecture resNet.

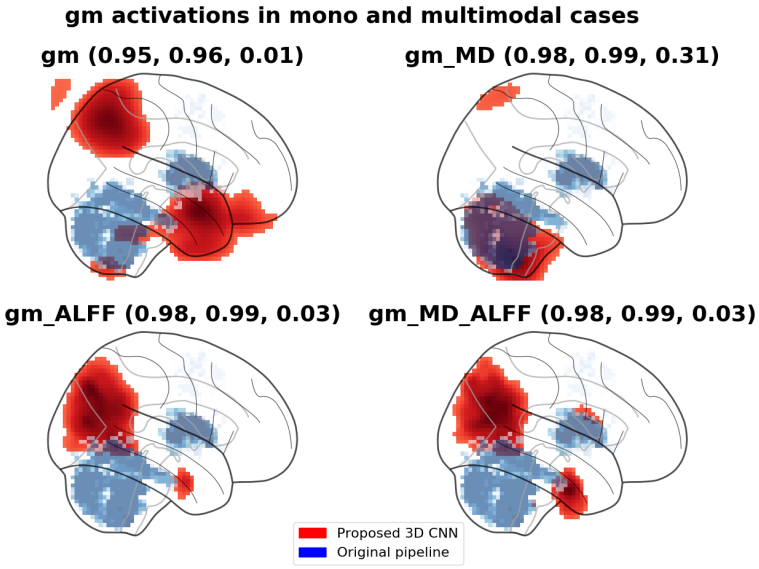


FIGURE 5.14: Visualisation des zones discriminantes avec la méthode CNN eyes visions pour le biomarqueur GM (rouge) et comparaison avec la méthode F. Nemmi et. al [46] (bleu)

Seule la combinaison des biomarqueurs GM et MD obtient des voxels discriminants en commun avec F. Nemmi et. al. Ceci correspond aux scores obtenus (voir table 5.2) puisque seules les combinaisons des biomarqueurs MD ainsi que GM et MD ont une sensitivité élevée et sont donc capable de détecter les patients AMS. Autrement dit, seules ces combinaisons apprennent les caractéristiques correspondant aux patients AMS.
Le cervelet est alors mis en évidence alors que le putamen est omis.

La figure 5.16 montre les visualisations des voxels discriminants du biomarqueur ALFF avec le modèle basé sur l'adaptation 3-dimensions de l'architecture resNet.
Les scores de sensitivité sont faibles sur toutes les combinaisons des biomarqueurs et les modèles ne sont donc pas performants pour détecter les caractéristiques des patients AMS, ce qui se ressent sur les visualisations.

5.4.3 Visualisation de l'adaptation 3-dimensions du modèle GoogleNet

Les modèles basés sur l'adaptation 3-dimensions du modèle googleNet sont plus profonds que les modèles basés sur VGG. Tout comme pour le modèle ResNet, il y a plus de décompositions intermédiaires et les effets indésirables de l'interpolation sont plus présents que pour le modèle VGG.

La visualisation des voxels discriminants obtenus avec le modèle GoogLeNet sur le biomarqueur GM est présentée figure 5.17. Seule la combinaison GM et MD permet de cibler le cervelet et d'obtenir des voxels discriminants en commun avec F. Nemmi et. al [46]. Le putamen est considéré comme non-discriminant par le modèle GoogLeNet.

La visualisation des voxels discriminants obtenus avec le modèle GoogLeNet sur le biomarqueur MD est présentée figure 5.18. Le biomarqueur MD seul et la combinaison GM et MD permet de

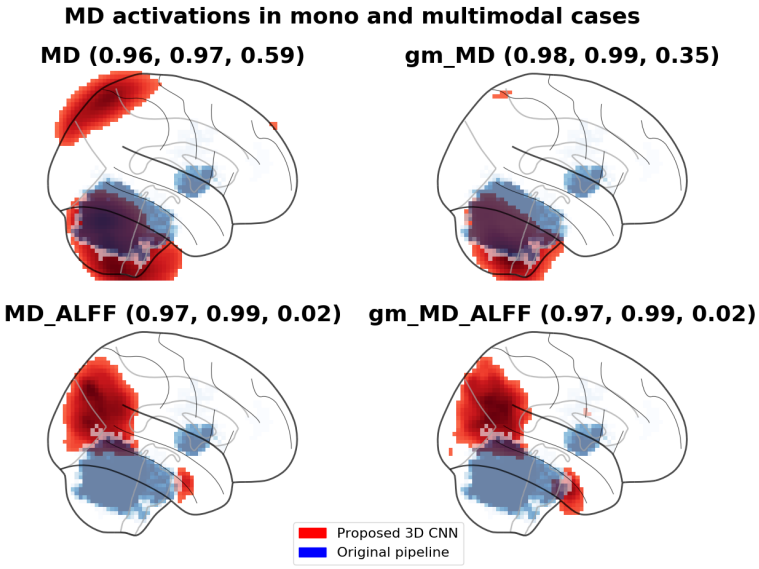


FIGURE 5.15: Visualisation des zones discriminantes avec la méthode CNN eyes visions pour le biomarqueur MD (rouge) et comparaison avec la méthode F. Nemmi et. al [46] (bleu)

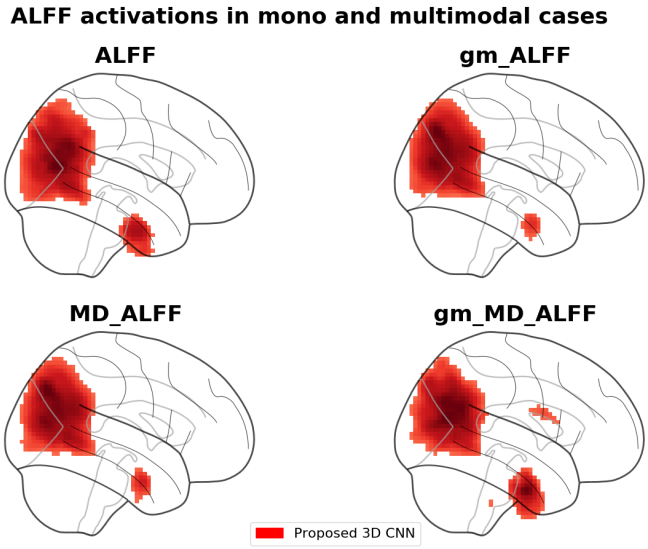


FIGURE 5.16: Visualisation des zones discriminantes avec la méthode CNN eyes visions pour le biomarqueur ALFF

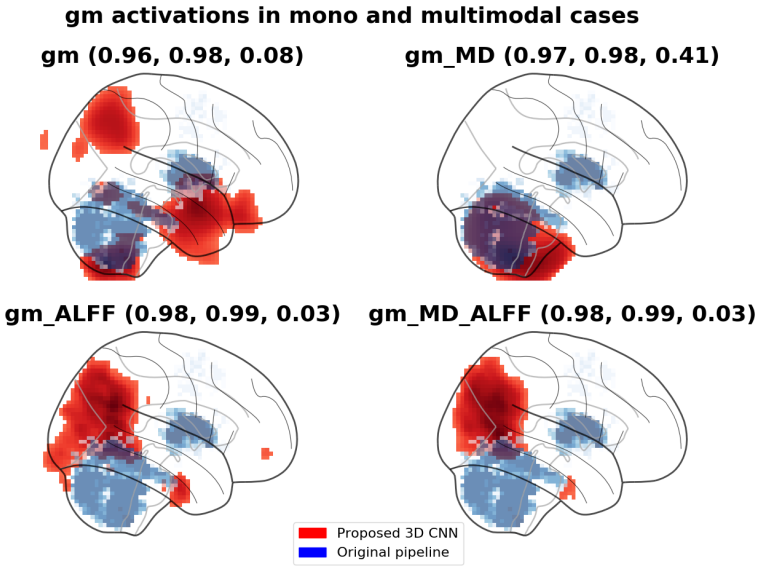


FIGURE 5.17: Visualisation des zones discriminantes avec la méthode CNN eyes visions pour le biomarqueur GM (rouge) et comparaison avec la méthode F. Nemmi et. al [46] (bleu)

cibler le cervelet et d'obtenir des voxels discriminants en commun avec F. Nemmi et. al [46]. Tout comme pour le biomarqueurs GM, le putamen est omis des voxels discriminants.

Les voxels discriminants obtenus pour le biomarqueur ALFF sont présentés sur la figure 5.19. Le biomarqueur ALFF seul, et la combinaison GM et ALFF possèdent une sensitivité faible et ne sont donc pas capables de détecter les caractéristiques des patients AMS, ce qui se ressent sur les visualisations. Les combinaisons de biomarqueurs MD et ALFF ainsi que GM, MD et ALFF ont de meilleurs scores de sensitivité, cependant les visualisations sont similaires au biomarqueur ALFF seul et la combinaison GM et ALFF. Cela laisse penser que les scores de sensitivité sont meilleurs de part la présence du biomarqueur MD.

5.4.4 Analyse des visualisations des voxels discriminants

Les modèle des trois architectures ont été entrainés en suivant le même schéma, sur les mêmes données et avec la même découpe de validation croisée. Cependant les visualisations obtenues sont différentes selon les architectures. Les visualisations calculées sur l'architecture VGG sont plus précises alors que celles issues des architectures ResNet et GoogLeNet sont très lissées.

L'une des raisons est le fait que ResNet et GoogLeNet utilisent plus de couches de convolution. Ainsi, en fusionnant toutes les décompositions intermédiaires, la méthode de visualisation prend en compte plus de visualisations où les données sont de très petites dimensions. Les effets d'interpolation sont alors plus présents que pour l'architecture VGG.
Un correctif visant à pondérer les décompositions intermédiaires selon la profondeur de la couche courante pourrait réduire cet effet.

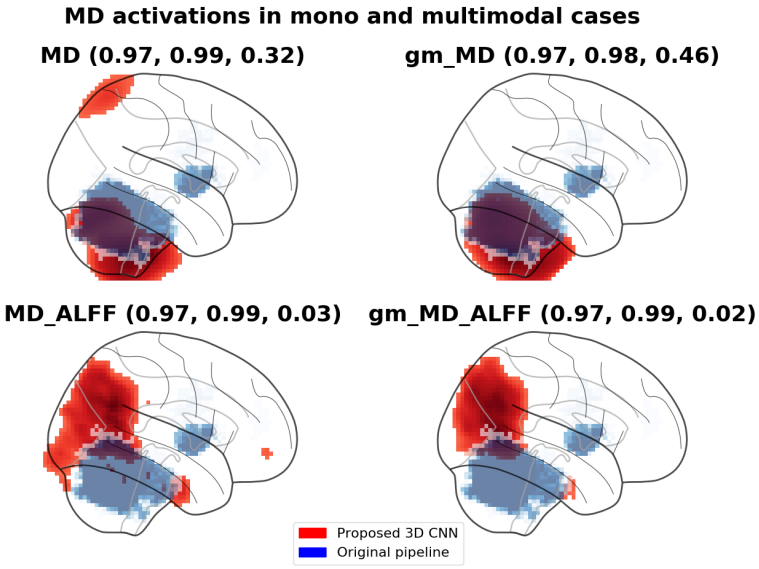


FIGURE 5.18: Visualisation des zones discriminantes avec la méthode CNN eyes visions pour le biomarqueur MD (rouge) et comparaison avec la méthode F. Nemmi et. al [46] (bleu)

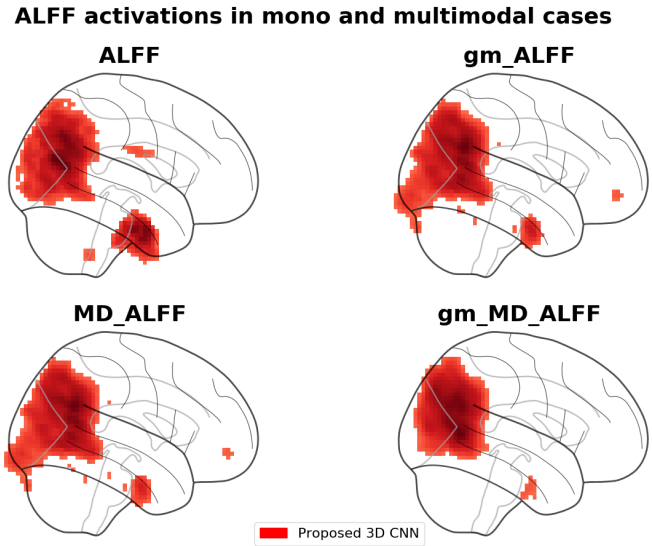


FIGURE 5.19: Visualisation des zones discriminantes avec la méthode CNN eyes visions pour le biomarqueur ALFF

Une autre raison serait l'utilisation du "zero-padding" sur les modèles ResNet et GoogLeNet. L'ajout de ces zéros réduit la proportion d'information utile dans l'image mais reste indispensable dans la conception de ces architectures, afin de pouvoir appliquer les couches de concaténation et d'addition qui nécessitent des dimensions égales.

5.5 Applications cliniques

Le pipeline basé sur le "deep learning" multimodal a pu être appliqué avec succès à l'atrophie multi systématisée, un syndrôme Parkinsonien rare. Il a été développé de façon à pouvoir comparer les résultats avec l'étude publiée de F. Nemmi et. al [46]. Or le groupe de F. Nemmi et ses collègues ont pu appliquer leur pipeline basé sur les Machines à vecteur de support à d'autres pathologies neurodégénératives. En particulier la neurofibromatose [48, 65] mais aussi pour mesurer l'intégrité structurelle et fonctionnelle du réseau frontopariétal chez les patients victimes de lésions cérébrales traumatiques et de coma anoxo-ischémique [50].

Les résultats obtenus dans cette étude basée sur le "deep learning" étant comparables à ceux obtenus par F. Nemmi et. al, ce pipeline de réseaux de neurones convolutifs 3-dimensions multimodaux pourrait alors être appliqué à d'autres pathologies neurodégénératives. De plus, le biomarqueur d'amplitude de fluctuation des basses fréquences s'est révélé discriminant alors qu'il n'était pas retenu comme tel dans le pipeline de F. Nemmi et. al. Ceci offre alors de nouvelles perspectives quant à l'utilisation des réseaux de neurones pour les pathologies neurodégénératives.

De plus cette étude sur l'atrophie multi systématisée a été réalisée en collectant 29 sujets sains et 26 patients AMS. Le jeu de donnée était ici limité de par la difficulté à rassembler un grand nombre de patients atteints d'une pathologie rare. Cependant cela permet d'envisager d'employer ce pipeline de "deep learning" en routine clinique. En effet, un centre hospitalier pourrait constituer une base de données de sujets sains à comparer avec une base de données de toute autre pathologies neurodégénératives. Un jeu de données de l'ordre de la trentaine de patients est envisageable pour un centre hospitalier ou une clinique.
Le pipeline de "deep learning" permettrait alors à une clinique d'obtenir le pouvoir discriminant des biomarqueurs dérivés de l'IRM, ainsi que les zones du cerveau incriminées dans la pathologie étudiée. Cela pourrait aussi permettre d'éviter les examens cliniques qui ne sont pas retenus comme discriminants pour la pathologie courante. En effet ALFF s'est montré comme le biomarqueur le moins discriminant pour les patients atteints de l'AMS, et n'améliore pas significativement les résultats dans le cas multimodal. Ainsi, un praticien pourrait se passer de l'acquisition de l'IRM fonctionnelle, la plus couteuse en temps.

Conclusion

Ces travaux de thèse sur l'utilisation de l'intelligence artificielle pour l'aide au diagnostic des pathologies neurodégénératives visaient à répondre aux questions suivantes :

— Le deep learning permet-il d'obtenir une discrimination entre les patients atteints de pathologies neurodégénératives et les sujets sains en utilisant un jeu de données de l'ordre de la dizaine de patients ?

— Le deep learning permet-il d'analyser le pouvoir discriminant des biomarqueurs dérivés de l'IRM ?

— Le deep learning permet-il de définir une signature spatiale de la pathologie étudiée ?

En utilisant un jeu de données simulant une anormalité dans une zone du cerveau, il a été possible d'étudier la capacité de discrimination d'un réseau de neurones convolutifs 3-dimensions s'appuyant sur un jeu de données de petite taille.
Il a été montré dans la section 4.2 qu'à partir d'un certain seuil de différence entre les deux classes, un modèle de deep learning était capable de discriminer les sujets sains des sujets anormaux, malgré un jeu de données de petite taille compatible avec une application clinique.
De plus, l'application sur des patients atteints de pathologies neurodégénératives rares dans la section 5.2 a montré qu'il était possible de discriminer les patients atteints d'AMS des sujets sains avec un modèle de réseau de neurones convolutifs 3-dimensions en obtenant des scores d'exactitude de l'ordre de 90 % et un accord entre la sensitivité et la spécificité.

La section 5.3 a permis d'effectuer une analyse du pouvoir discriminant des différents biomarqueurs dérivés de l'IRM sur l'AMS. Il a alors été montré que l'imagerie fonctionnelle n'apportait pas d'informations supplémentaires sur la discrimination de l'AMS. Deux points ont étés alors mis en évidence via l'analyse des pouvoirs discriminants des biomarqueurs dérivés de l'IRM. Suite à une étude préliminaire, il est possible de limiter les examens cliniques aux biomarqueurs discriminants uniquement. Dans le cas de l'AMS, l'examen d'imagerie fonctionnelle n'est pas nécessaire pour la discrimination. Or cet examen d'imagerie fonctionnelle étant le plus couteux en temps n'étant pas nécessaire, une clinique peut alors organiser au mieux le planning d'utilisation de l'IRM ce qui réduit par la même occasion les coûts financiers.
De plus, chaque biomarqueur visant à imager la structure ou encore le fonctionnement du cerveau,

l'analyse du pouvoir discriminant permet d'orienter la recherche fondamentale des pathologies neurodégénératives vers une cause et donc une meilleure compréhension de ces pathologies.

En développant une méthode de visualisation des voxels discriminants détaillée dans la section 3.5 particulièrement adaptée à la neuro imagerie 3-dimensions, il a été montré qu'il était possible de calculer une signature spatiale d'une pathologie neurodégénérative.
L'analyse des zones discriminantes et du pouvoir discriminant des biomarqueurs dérivés de l'IRM permet une meilleure compréhension des pathologies neurodégénératives et des processus amenant un patient à déclarer une pathologie.

Finalement, plusieurs contributions sont apportées via ces travaux de thèse. Une méthode de visualisation des zones discriminantes et son logiciel associé, une analyse des pouvoirs discriminants des différents biomarqueurs dérivés de l'IRM et la discrimination des patients pathologiques des sujets sains tout en utilisant un jeu de données de petite taille compatible avec une application clinique.
Ces travaux permettent d'envisager une meilleure compréhension des pathologies neurodégénératives, une utilisation dans les routines cliniques mais aussi une étude sur les pathologies idiopathiques.

Perspectives

L'application avec succès à l'AMS où il est possible d'extraire de l'état de l'art à la fois les scores de discrimination, le pouvoir discriminant et les zones du cerveau discriminantes permet d'envisager ces travaux de thèse sur d'autres pathologies neurodégénératives et ainsi améliorer la compréhension de ces pathologies mais aussi leur diagnostic.

Afin d'envisager une application à d'autres pathologies neurodégénératives, il serait nécessaire de constituer un jeu de données de biomarqueurs dérivés de l'IRM sur la pathologie étudiée. Cette étape peut déjà être un challenge pour les pathologies rares où la collecte de données est difficile, ou encore pour les pathologies idiopathiques dont le diagnostic peut parfois être incertain.

Une autre perspective de ces travaux de thèse serait d'envisager une utilisation de ce pipeline en combinant plusieurs centre de recherche ou cliniques. En effet, chaque centre disposant de sa machine IRM avec ses propres caractéristiques, la fusion de plusieurs jeu de données pourrait être problématique. Une autre possibilité d'utilisation multi-centre serait d'employer le transfer learning. Un centre entrainerait alors un modèle en entier et partegerait le modèle entrainé à d'autres centres qui eux n'entraineraient que les couches de prédiction.

L'application de méthode de data augmentation pourrait améliorer les résultats obtenus, cependant l'utilisation de telles méthodes en neuro imagerie reste difficile. Les travaux de thèse de Giulia Maria Mattia sur la simulation de données de neuro imagerie pourraient apporter des éléments de réponses. En effet si la version de données simulées présentée dans ce manuscrit de thèse vise à simuler une anormalité sur une zone précise du cerveau, ses travaux de thèse s'orientent vers la simulation de l'IRM d'un cerveau pathologique. Ses travaux pourraient alors donner lieu à une méthode de data augmentation compatible avec la neuro imagerie.

Pour finir, sur un plus long terme, la création d'une plateforme d'aide au diagnostic des pa-

thologies neurodégénératives pourrait être envisagée. En effet, en combinant le pipeline basé sur le deep learning présenté dans ce manuscrit de thèse avec les travaux de thèse de Giulia Maria Mattia pour l'augmentation de données ainsi que des travaux sur une utilisation multi-centres, une plateforme fournissant aux praticiens un outil prêt à l'emploi d'aide au diagnostic des patients atteints de pathologies neurodégénératives pourrait être développée.

Annexes

Publications

PUBLICATIONS

- **E. Villain**, F. Nemmi, A. Pavy Le Taron, O. Rascol, X. Franceries, P. Péran, and M.-V. Le Lann
  Multiple System Atrophy diagnosis using 3-dimension multi-modal Convolutional Neural Networks
  Toulouse, France, pp. 69-71
  https://hal.archives-ouvertes.fr/hal-02161172
  Rencontre des Jeunes Chercheurs en Intelligence Artificielle (RJCIA) - Juillet 2019

- **E. Villain**, F. Nemmi, A. Pavy Le Taron, O. Rascol, X. Franceries, P. Péran, and M.-V. Le Lann
  Convolutional neural network for discriminating between Multiple System Atrophy and Healthy Control, comparing MRI modalities and highlighting the disease signature
  vol. 35. S1, pp. 120-121. DOI: https://doi.org/10.1002/mds.28268
  Movement Disorder - 12-16 Septembre 2020

- **E. Villain**, G. M. Mattia, F. Nemmi, P. Péran, X. Franceries and M. V. le Lann,
  Visual interpretation of CNN decision-making process using Simulated Brain MRI, 2021
  IEEE 34th International Symposium on Computer-Based Medical Systems (CBMS),
  7-9 Juin 2021, pp. 515-520, doi: 10.1109/CBMS52027.2021.00102.

- G. Sidorski, J. Mazurier, I. Berry, X. Franceries, **E. Villain**, B. Pichon, B. Pinel, G. Jimenez, O. Gallocher, C. Chevelle, D. Marre, J. Camilleri, V. Connord, Y. Marty, N. Mathy, D. Zarate, I. Latorzeff
  Génération automatique de plans de traitements en radiothérapie externe : Apport de l'intelligence Artificielle dans les cancers de la prostate
  DOI : https://doi.org/10.1016/j.canrad.2021.07.019
  Société Française de radiothérapie Oncologie (SFRO) - p 735-736 - 6-8 Octobre 2021

- Mattia, Giulia Maria; Nemmi, Federico; **Villain, Edouard**; Le Lann, Marie-Véronique; Franceries, Xavier; Péran, Patrice (2021):
  Investigating the Discrimination Ability of 3D Convolutional Neural Networks Applied to Altered Brain MRI Parametric Maps.
  IEEE TechRxiv. Preprint. https://doi.org/10.36227/techrxiv.15010803.v1

PUBLICATIONS ACCEPTEES

- G. Maria Mattia, **E. Villain**, F. Nemmi, O. Rascol, W-G. Meissner, X. Franceries, P. Péran
  Neurodegenerative traits detected via 3D CNNs trained with simulated brain MRI : Prédiction supported by visualisation of discriminant voxels
  (Accepté) IEEE International Conference on Bioinformatics and Biomedecine (BIBM) – 9-12 Décembre 2021

PUBLICATIONS SOUMISES

- **E. Villain**, G. Maria Mattia, F. Nemmi, A. Pavy Le Taron, O. Rascol, W-G. Meissner, X. Franceries, P. Péran, M-V. Le Lann
  3D Convlutional Neural Network using multimodal MRI for Multiple System Atrophy classification
  Soumis - Frontiers in Artificial Intelligence - 2021

- G. Maria Mattia, **E. Villain**, O. Rascol, W-G. Meissner, X. Franceries, P. Péran
  Multiple System Atrophy Classification via 3D Convolutional Neural Network and Simulated Brain MRI Parametric Maps
  soumis - International Society for Magnetic Resonance in Medecine (ISMRM) - 2021

Articles visualisation des voxels discriminants

Visual interpretation of CNN decision-making process using Simulated Brain MRI

Edouard VILLAIN
LAAS CNRS, Université de Toulouse, CNRS, INSA, UPS
Inserm ToNIC, UMR 1214
Toulouse NeuroImaging Center
Toulouse, France
evillain@laas.fr

Giulia Maria MATTIA
Inserm ToNIC, UMR 1214
Toulouse NeuroImaging Center
Toulouse, France
giulia-maria.mattia@inserm.fr

Federico NEMMI
Inserm ToNIC, UMR 1214
Toulouse NeuroImaging Center
Toulouse, France
federico.nemmi@inserm.fr

Patrice PÉRAN
Inserm ToNIC, UMR 1214
Toulouse NeuroImaging Center
Toulouse, France
patrice.peran@inserm.fr

Xavier FRANCERIES
Inserm CRCT, UMR 1037
Centre de Recherche en Cancérologie de Toulouse
Toulouse, France
xavier.franceries@inserm.fr

Marie Véronique LE LANN
LAAS CNRS, Université de Toulouse, CNRS, INSA, UPS
Toulouse, France
mvlelann@laas.fr

***Abstract*—Convolutional neural networks (CNNs) are being extensively used to analyze medical images given the remarkable performances achieved so far. Due to the non-transparent decision-making process, CNNs are thought to be black boxes, so hindering their applicability. We submit a novel visualization technique to shed light on CNNs decisions in a classification task. Brain magnetic resonance images are fed as input to an original 3D CNN to allow discrimination of normal against modified brain data. This modification targets specific brain regions by linearly increasing their intensity, and involves regions with very different features in dimension, position, and enclosed tissues. The proposed visualization method merges all convolutional layers output in order to highlight where the model is "looking" during the decision-making process. Our visualizations allow to recover the same areas modified in the images, thus proving they are relevant to the prediction as expected. Comparing results from models with different accuracy, show that even in the case of low performance the expected regions are present in the activation maps leading the way to ameliorations of the CNN architecture.**

Keywords—CNN, Brain MRI, visual interpretation, simulated dataset

I. Introduction

One of deep learning most successful tools, convolutional neural networks (CNNs) are more and more used in medical image analysis due to the excellent performance achieved on natural image recognition [1]-[3]. Composed of multiple layers, CNNs can indeed produce representations from multidimensional arrays with different abstraction levels [4].
In the neuroimaging community, magnetic resonance imaging (MRI) can be found among the preferred imaging techniques to non-invasively investigate brain functions and structure, often fed as input to convolutional neural networks, i.e. for classification and segmentation tasks [5]-[6].

Although powerful and widely employed, CNNs are referred to as black boxes because of the opaque decision making process, thus impeding their acceptability and usage [7]. Regardless of the application, one of the major challenges in deep learning is associating outstanding performances with convincing and exhaustive explanations [8] which can be provided at the processing level (e.g. LIME [9], Grad-CAM [10], saliency maps [9]), by creating representations referring to their subcomponents [11], or designing systems able to produce their own explanations [12]. To cope with this aspect, diverse techniques have been conceived to enable a finer although marginal (as restricted to specific components of CNNs) understanding of their behavior [13]. Especially in the medical field, providing solid interpretation methods is of paramount importance to allow physicians and medical practitioners to understand the reasons and the process behind neural networks outcome [14]. Various attempts have been made so far in the neuroimaging domain to address this need. Saliency maps were exploited in [15] to highlight salient features for the classification of autism patients with an ensemble strategy using a 3D CNN model. Deconvolution visualization technique was employed in [16] to discover relevant areas for distinguishing Parkinson's Disease (PD) patients from normal controls by means of a 3D CNN, obtaining brain heatmaps with the occlusion technique [17]. Identification of key regions in brain tumor segmentation was performed in [18] with Grad-CAM and compared according to different CNN architectures, whereas the authors in [19] developed a pyramidal structure for the network to combine Grad-CAM visualizations at diverse scales. An extension of the technique in [20] was designed including 3D brain masks that covered significant parts of the images for correct classification of Alzheimer's Disease (AD) patients via a 3D CNN [15].
In this study, we submit a novel visualization technique, based on the output of convolutional layers, to discover crucial areas

in the discrimination of normal against modified mean diffusivity (MD) maps using an original 3D CNN. Mean diffusivity is a measure of water diffusion derived from diffusion tensor imaging (DTI) of the brain [21]. It has been used to track white and grey matter degeneration in several neurodegenerative diseases [22]-[23]. We introduced region-specific abnormalities to MD maps by linearly altering their intensity. We will refer to these images as abnormal-induced (AbIn). Two regions have been taken into account for the modification, i.e. cerebellum and putamen, as they differ in position, size, and tissue. Using the proposed visualization method, we were able to retrieve through the activation maps the most relevant areas for the discrimination, i.e. the brain regions which had been altered. In addition, we compared visualizations from models with high and low accuracy: models with better accuracy provided clearer visualizations, although models with poor performances presented more noisy visualizations still containing the desired meaningful regions.

II. Material and Method

A. Abnormal-induced MRI

1) Dataset: A 3-T MRI machine (Philips Achieva, Inserm/UPS UMR1214 ToNIC Technical Platform, Toulouse, France) and a head antenna with 32 channels were used to scan 89 subjects, all males aged between 20.7 and 85.3 years (mean age = 56.2 years, standard deviation = 18.1 years). Diffusion-weighted images were acquired with parameters: b-value (number of directions) = 0 (1); 500 (32); 1000 (32) s/mm^2; TE = 55 ms; TR = 12:36 s.

The standard FSL pipeline [24] was exploited to process these images, from which mean diffusivity (MD) maps were obtained, after non-linear registration in MNI space with a resolution of 3 mm isotropic.

2) Image creation: We produced abnormal-induced MD by means of a linear transformation of intensity to specific regions belonging to normal MD maps. Among the various brain regions, we selected cerebellum and putamen as they are characterized by different dimension, position and type of cerebral tissues.

Each region (denoted by j in (1)) isolated using an atlas-based mask [25] was modified as follows:

$$w_j = (i + 1) \cdot z_j \quad (1)$$

with w_j represents the AbIn image and z_j the raw image, whereas i indicates the percentage for the intensity increase in the interval [3%, 99%] at steps of 3%. An intensity threshold set to the 75th and 90th percentiles respectively for cerebellum and putamen was applied to diminish saturation effects on the images.

Mono-region AbIn images are featured with only one abnormal region, whereas bi-region AbIn images were created via the abovementioned method applied singularly to each considered region using equal percentages for the intensity increase.

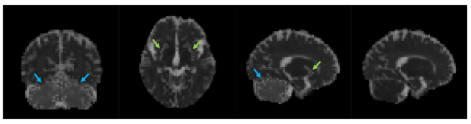


Figure 1: From left to right: cerebellum AbIn MD, putamen AbIn MD, Cerebellum/Putamen AbIn MD, raw MD map. AbIn MD were modified at 99% intensity increase. Blue arrows point to cerebellum, green arrows to putamen

B. CNN

The artificial neuron processes inputs to compute an output value with a simple equation that multiplies those inputs by its weights and add a bias term. The output is then modified by an activation function φ. The whole equation of the artificial neuron is detailed in (2) with x_i the inputs, t the output, ω_i and b the artificial neuron's weights and bias and φ the activation function.

$$t = \varphi\left(\sum_{i=0}^{n} x_i . \omega_i + b\right) \quad (2)$$

The training phase serves to set CNN weights and bias which minimize a loss function.

The artificial neural network is then created by connecting the output to the input of another artificial neuron. Most architectures based on artificial neurons are organized in layers where outputs from a layer are fully connected (FC) to all the inputs of the next layer. Additional details about neural networks can be found in [26].

Convolution neural networks allow processing very large-scale data (e.g. images) by connecting layers no longer with just weights but with convolution filters, whose weights are determined during training. The last part of the CNN for a classification task comprises flattening outputs from the last convolution layer to be then used by FC layers. The first part of the CNN consists in the convolution part of the CNN which extracts relevant patterns of the input whereas the FC one computes the prediction based on the extracted patterns.

In this work, a 3D adaptation of VGG16 architecture [27] is proposed and detailed in fig. 2. Convolutional (fig. 2 (a)) and FC layers (fig. 2 (b)) are followed by batch normalization and exponential linear unit (ELU) activation function. Batch normalization layers can reduce the effect of the covariance shift on the training [28], whereas ELU activation accelerates the training phase [29]. The last FC layer contains a neuron for each class of the classification task and is followed by a SoftMax activation layer that transform outputs in membership probabilities (fig 2 (c)). The CNN prediction outputs the classes with the maximum membership probabilities. Pooling layers scale down dimensions by preserving a unique value in a small subset of units belonging to the previous layers. The final architecture is composed of a sequence of convolution layers with a (5,5,5) kernel size and average pooling layers, followed by FC layers (fig. 2 (d)).

Models are then trained with categorical cross entropy as loss function using Adam optimizer [30] with an initial learning rate of 10^{-5}.

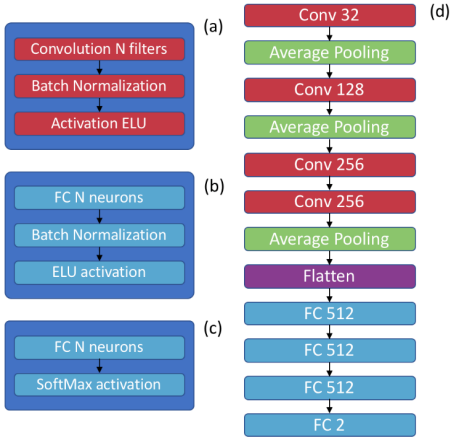


Figure 2: Detail of the CNN architecture (d) with convolution layers (a) Fully Connected (FC) layers (b) and prediction layer (c). N is the number of filters or neurons as indicated

C. Visualization

In order to get a visual interpretation of the CNN prediction process, a simple and fast method was developed to compute an activation map. Similarly to GradCAM [10] or saliency map [9], the main goal is to establish which part of the input is the most relevant to the prediction.

To this aim, we collect each output of every convolution layer for every filter and for each set of data separately. A thresholding is then applied in order to remove negative values as they do not provide relevant information. A bicubic interpolation is performed to match input dimensions. Then outputs of each filter are merged by computing the mean, so as to obtain a single activation map by convolution layers. Finally, activation maps of each convolution layer are also merged by a mean operation followed by normalization. The result is a single activation map for each example of the dataset. The final step for the visual interpretation is to join all activation maps by class, separately for training and testing sets, and computing the absolute difference between the mean activation map of both classes.

D. Benchmark

A benchmark was defined and composed of three steps to obtain the presented results:

1. generation of simulated data, i.e. abnormal-induced MD maps;
2. training of the models;
3. computation of activation maps.

These steps are repeated for each set of abnormal-induced MD, each with a specific intensity increase, providing a complete display of the performances of the proposed visualization method, with different input data and model fitting.

III. Results

The following results present the activation maps obtained for mono and bi-region AbIn MD on training and testing sets, for a model with high accuracy (1.0) and low accuracy (~ 0.65). Section III.C provides results computed with a single subject as an example of a clinical application.

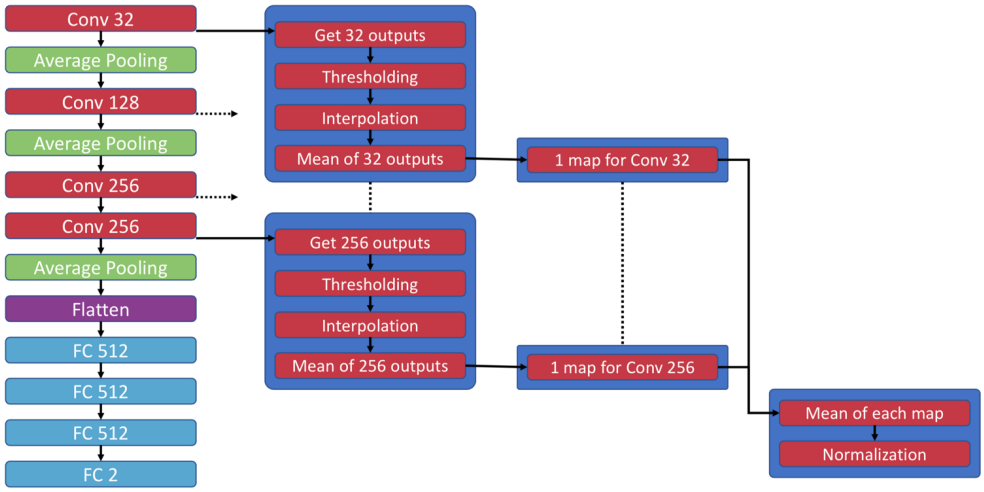


Figure 3: Detail of the proposed visualization method

This study was sponsored by Inserm and funded by a "Recherche clinique translationnelle" grant from INSERM – DGOS (2013 – 2014)

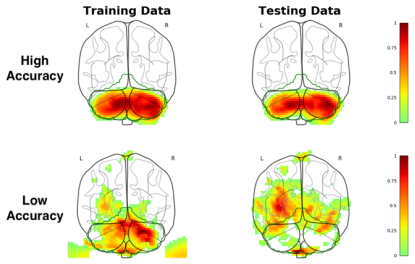


Figure 4: Visualization of the absolute distance between the activation maps from both classes of cerebellum abnormal-induced data for training (left) and testing (right) dataset for a model with high accuracy (up) and low accuracy (down). The green contour represents cerebellum mask

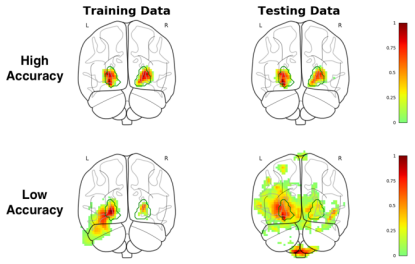


Figure 5: Visualization of the absolute distance between the activation maps from both classes of putamen abnormal-induced data for training (left) and testing (right) dataset for a model with high accuracy (up) and low accuracy (down). The green contour delineates putamen mask

A. Mono-region results

Fig. 4 shows the absolute difference between the mean activation maps of the original data (class 0) and the abnormal-induced class (class 1) for cerebellum. The upper part of fig. 4 is computed for a model with 1.0 accuracy on training and testing sets. The produced visualization highlights the inner part of cerebellum, which is indeed the one subjected to the intensity increase. The lower part of fig 4. displays visualizations computed for a model with low accuracy (~ 0.65), in which irrelevant voxels are activated. These phenomena are more visible on the testing set since these data had not been used during the training phase. Even if the visual interpretation is noisier with a low fitting model, the cerebellum is however activated for the training set.

Fig. 5 shows the visualizations for the putamen abnormal-induced images. The visual interpretation clearly targets the putamen for the model with high accuracy whereas it appears degraded for the model with low accuracy. A low activation on the training set was observed and many irrelevant voxels on the testing set.

Mono-region visual interpretations target the area of interest and not surprisingly are more accurate for models with high accuracy.

B. Bi-region results

Fig. 6 presents the absolute difference between the mean activation maps of the original data class and the cerebellum and putamen abnormal-induced class. Once again, the model with high accuracy obtained better visual interpretation than model with low accuracy.

The putamen area is not clearly activated in favor of the cerebellum area.

C. Single-subject application

In addition to the activation maps in section III.A and III.C on the entire dataset aiming to analyze a global visual interpretation, the proposed visual interpretation was studied for a single subject application. Fig. 7 shows the absolute

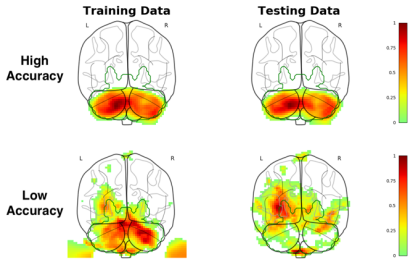


Figure 6: Visualization of the absolute distance between the activation maps from both classes of cerebellum and putamen abnormal-induced data for training (left) and testing (right) dataset for a model with high accuracy (up) and low accuracy (down). The green line contours putamen and cerebellum areas

difference between the activation maps considering original data of a random subject from the testing dataset and the mean activation map of both classes. The upper part of these figures was computed for the normal images whereas the lower part concerns an abnormal-induced subject and this for each modified brain area. Visual interpretations computed for a single subject application on the cerebellum and putamen abnormal-induced data in fig. 7 suggest that original data from a single subject display more differences with the mean activation map of the abnormal-induced one and vice versa.

Indeed, the visual interpretation enables to both look at the model prediction and the most discriminant voxels.

Since putamen has a small size compared to cerebellum, we noticed that results are not so straightforward. We also noticed that some areas are activated on both visual interpretations. These findings are not unexpected since each subject has peculiar traits, and the visual interpretation can detect some areas that are different from the mean activation of both classes.

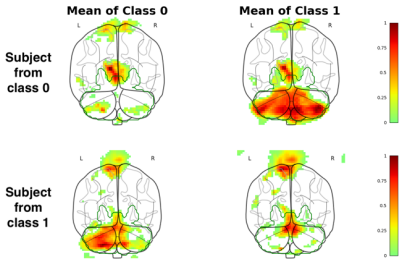


Figure 7: Visualization of the absolute distance between the activation maps from a random subject of both classes and the mean activation map on the training dataset for cerebellum and putamen abnormal-induced data. The green line contours putamen and cerebellum areas

Nevertheless, these undesired areas make the visual analysis less interpretable, particularly when the difference between classes is negligible. Considering bi-region AbIn images, we obtained comparable results as those mentioned previously: cerebellum is in fact much more dominant with respect to putamen. Moreover, we can clearly see that a single-subject activation map presents more dissimilarities with the mean activation map from AbIn images and reciprocally.

IV. Discussion

In this study, a visualization method to compute an activation map providing the most discriminant voxels for classification with a 3D CNN of brain magnetic resonance images was devised. Utilizing the AbIn simulated set allowed to compare the visualization with the ground truth on a plausible dataset. Indeed, similarly to a real-world discrimination problem on brain imaging, subjects within a class share some relevant modifications, but each image also suffers from acquisition noise and individual variability related to physiological differences among subjects. Some studies proposed different methods in order to compute an activation map showing the most discriminant parts used during the decision making process of neural networks such as GradCAM, [10] or saliency map [9], but they are characterized by a long computation time compared to the proposed visualization method. Indeed, even with 3D input such as those used in this study, an activation map is obtained in less than a second. The proposed method was able to find relevant areas to the CNN decision making process, for different regions (cerebellum and putamen) and intensity modification. It is also remarkable that the proposed method provides some interesting activation maps on models with low accuracy (∼0.65). Even if the results obtained with low accuracy models are degraded, the region of interest is still targeted. This can help to conduct a preliminary investigation to refine the current architecture and improve prediction, something that could be extremely useful in a clinical application with a few subjects on little-known diseases. We also posit a single subject application of the visualization method, offering interesting insights on a subject-level, also from a clinical point of view.

The proposed visualization technique was also utilized on real multiple system atrophy patient in [31] and managed to localize cerebellum and putamen, key areas in line with the current knowledge of the disease [32]. It is worth noticing that the proposed method can be applied to all architectures present in the state-of-the-art, both for 2 and 3-dimensional data, as well as in multimodal cases, e.g. with multiple type of input data. Another interesting advantage of the proposed method is that only convolutional layers are used. This could reveal beneficial for transfer learning since activation maps of different pre-trained model can be computed, before retraining the FC layers, allowing the user to make a more informed choice on the optimal architecture for his problem by choosing the least diffuse, most intense visualization or by using a priori information on the expected results.

In our future work, we will compare this new visualization method with gradCAM [10].

Acknowledgment

The study was sponsored by Inserm and funded by a “Recherche clinique translationnelle” grant from INSERM-DGOS (2013-2014). The authors declare no conflict of interest.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, Jun. 2017, doi: 10.1145/3065386.

[2] G. Litjens *et al.*, “A Survey on Deep Learning in Medical Image Analysis,” *Med. Image Anal.*, vol. 42, pp. 60–88, Feb. 2017, doi: 10.1016/j.media.2017.07.005.

[3] D. Shen, G. Wu, and H. Il Suk, “Deep Learning in Medical Image Analysis,” *Annu. Rev. Biomed. Eng.*, vol. 19, pp. 221–248, Jun. 2017, doi: 10.1146/annurev-bioeng-071516-044442.

[4] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[5] S. M. Plis *et al.*, “Deep learning for neuroimaging: a validation study,” *Front. Neurosci.*, vol. 8, no. 8 JUL, p. 229, Aug. 2014, doi: 10.3389/fnins.2014.00229.

[6] S. S. M. Salehi, D. Erdogmus, and A. Gholipour, “Auto-context Convolutional Neural Network (Auto-Net) for Brain Extraction in Magnetic Resonance Imaging,” *IEEE Trans. Med. Imaging*, vol. 36, no. 11, pp. 2319–2330, Mar. 2017, Accessed: Feb. 22, 2021. [Online]. Available: http://arxiv.org/abs/1703.02083.

[7] A. Shrikumar, P. Greenside, A. Shcherbina, and A. Kundaje, “Not Just a Black Box: Learning Important Features Through Propagating Activation Differences,” *arXiv*, vol. 1, pp. 0–5, May 2016, Accessed: Feb. 15, 2021. [Online]. Available: http://arxiv.org/abs/1605.01713.

[8] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, “Explaining Explanations: An Overview of Interpretability of Machine Learning,” *Proc. - 2018 IEEE 5th Int. Conf. Data Sci. Adv. Anal. DSAA 2018*, pp. 80–89, May 2018, Accessed: Feb. 15, 2021. [Online]. Available: http://arxiv.org/abs/1806.00069.

[9] M. T. Ribeiro, S. Singh, and C. Guestrin, “‘Why should i trust you?’ Explaining the predictions of any classifier,” in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery*

and Data Mining, Aug. 2016, vol. 13-17-August-2016, pp. 1135–1144, doi: 10.1145/2939672.2939778.

[10] R. Rs, M. Cogswell, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization," 2017, pp. 618–626, doi: 10.1109/ICCV.2017.74.

[11] Q. Zhang and S.-C. Zhu, "Visual Interpretability for Deep Learning: a Survey," *Front. Inf. Technol. Electron. Eng.*, vol. 19, no. 1, pp. 27–39, Feb. 2018, Accessed: Feb. 15, 2021. [Online]. Available: http://arxiv.org/abs/1802.00614.

[12] L. A. Hendricks, Z. Akata, M. Rohrbach, J. Donahue, B. Schiele, and T. Darrell, "Generating visual explanations," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016, vol. 9908 LNCS, pp. 3–19, doi: 10.1007/978-3-319-46493-0_1.

[13] D. Elton and D. C. Elton, "EasyChair Preprint Self-explaining AI as an alternative to interpretable AI Self-explaining AI as an alternative to interpretable AI," EasyChair, May 2020.

[14] A. Holzinger, C. Biemann, C. S. Pattichis, and D. B. Kell, "What do we need to build explainable AI systems for the medical domain?," *arXiv*, Dec. 2017, Accessed: Feb. 16, 2021. [Online]. Available: http://arxiv.org/abs/1712.09923.

[15] M. Khosla, K. Jamison, A. Kuceyeski, and M. R. Sabuncu, "Ensemble learning with 3D convolutional neural networks for functional connectome-based prediction," *Neuroimage*, vol. 199, pp. 651–662, Oct. 2019, doi: 10.1016/J.NEUROIMAGE.2019.06.012.

[16] S. Esmaeilzadeh, Y. Yang, and E. Adeli, "End-to-End Parkinson Disease Diagnosis using Brain MR-Images by 3D-CNN." 2018.

[17] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," Nov. 2013, Accessed: Nov. 19, 2019. [Online]. Available: http://arxiv.org/abs/1311.2901.

[18] P. Natekar, A. Kori, and G. Krishnamurthi, "Demystifying Brain Tumour Segmentation Networks: Interpretability and Uncertainty Analysis," *arXiv*, Sep. 2019, Accessed: Feb. 16, 2021. [Online]. Available: http://arxiv.org/abs/1909.01498.

[19] S. Lee, J. Lee, J. Lee, C.-K. Park, and S. Yoon, "Robust Tumor Localization with Pyramid Grad-CAM," May 2018, Accessed: Oct. 21, 2019. [Online]. Available: http://arxiv.org/abs/1805.11393.

[20] R. Fong and A. Vedaldi, "Interpretable Explanations of Black Boxes by Meaningful Perturbation," *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2017-October, pp. 3449–3457, Apr. 2017, doi: 10.1109/ICCV.2017.371.

[21] D. Le Bihan, "Diffusion MRI: What water tells us about the brain," *EMBO Mol. Med.*, vol. 6, no. 5, pp. 569–573, 2014, doi: 10.1002/emmm.201404055.

[22] P. Péran *et al.*, "MRI supervised and unsupervised classification of Parkinson's disease and multiple system atrophy," *Mov. Disord.*, vol. 33, no. 4, pp. 600–608, Apr. 2018, doi: 10.1002/mds.27307.

[23] S. B. Vos, D. K. Jones, B. Jeurissen, M. A. Viergever, and A. Leemans, "The influence of complex white matter architecture on the mean diffusivity in diffusion tensor MRI of the human brain," *Neuroimage*, vol. 59, no. 3, pp. 2208–2216, Feb. 2012, doi: 10.1016/j.neuroimage.2011.09.086.

[24] T. E. J. Behrens *et al.*, "Characterization and propagation of uncertainty in diffusion-weighted MR imaging," *Magn. Reson. Med.*, vol. 50, no. 5, pp. 1077–1088, Nov. 2003, doi: 10.1002/mrm.10609.

[25] A. Hammers *et al.*, "Three-dimensional maximum probability atlas of the human brain, with particular reference to the temporal lobe.," *Hum. Brain Mapp.*, vol. 19, no. 4, pp. 224–247, Aug. 2003, doi: 10.1002/hbm.10123.

[26] M. R. Mohammadi, S. A. Sadrossadat, M. G. Mortazavi, and B. Nouri, "A brief review over neural network modeling techniques," in *2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)*, 2017, pp. 54–57.

[27] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Sep. 2014, Accessed: Jun. 25, 2019. [Online]. Available: http://arxiv.org/abs/1409.1556.

[28] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry, "How Does Batch Normalization Help Optimization?," May 2018, Accessed: Jun. 25, 2019. [Online]. Available: http://arxiv.org/abs/1805.11604.

[29] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)," Nov. 2015, Accessed: Jun. 24, 2019. [Online]. Available: http://arxiv.org/abs/1511.07289.

[30] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Dec. 2014, Accessed: Jun. 24, 2019. [Online]. Available: http://arxiv.org/abs/1412.6980.

[31] E. Villain, F. Nemmi, A. Pavy-Le Traon, W. Meissner, O. Rascol M.V. Le Lann, X. Franceries, P. Péran, "Movement disorders," in *Convolutional neural network for discriminating between Multiple System Atrophy and Healthy Control, comparing MRI modalities and highlighting the disease signature*, 2020, p. 121, [Online]. Available: https://onlinelibrary.wiley.com/doi/epdf/10.1002/mds.28268.

[32] F. Nemmi *et al.*, "A totally data-driven whole-brain multimodal pipeline for the discrimination of Parkinson's disease, multiple system atrophy and healthy control," *NeuroImage Clin.*, vol. 23, p. 101858, 2019, doi: 10.1016/j.nicl.2019.101858.

Table des figures

Liste des tableaux

List of Algorithms

Bibliographie

[1] J. P. Ignizio, "A brief introduction to expert systems," *Computers & Operations Research*, vol. 17, no. 6, pp. 523–533, 1990.

[2] P. Jackson, *Introduction to Expert Systems*. USA : Addison-Wesley Longman Publishing Co., Inc., 3rd ed., 1998.

[3] T. K. Ho, "Random decision forests," in *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1, pp. 278–282 vol.1, 1995.

[4] N. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, pp. 175–185, 1992.

[5] Z. Zhang, "Introduction to machine learning : k-nearest neighbors," *Annals of Translational Medicine*, vol. 4, no. 11, 2016.

[6] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, (New York, NY, USA), p. 144–152, Association for Computing Machinery, 1992.

[7] V. N. Vapnik, *The nature of statistical learning theory*. Berlin : Springer-Verlag, 1995.

[8] N. Cristianini and E. Ricci, *Support Vector Machines*, pp. 928–932. Boston, MA : Springer US, 2008.

[9] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 2013.

[10] W. Mcculloch and W. Pitts, "A logical calculus of ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 127–147, 1943.

[11] F. Rosenblatt, "The perceptron : A probabilistic model for information storage and organization in the brain.," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

[12] J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 79, pp. 2554–8, 05 1982.

[13] Y. Lecun, "Une procédure d'apprentissage pour réseau à seuil asymétrique," *Proceedings of Cognitiva 85, Paris*, pp. 599–604, 1985.

[14] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning Representations by Back-Propagating Errors*, p. 696–699. Cambridge, MA, USA : MIT Press, 1988.

[15] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, p. 541–551, Dec. 1989.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv 1409.1556*, 09 2014.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[20] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014.

[21] K. R. T. Fink and J. R. Fink, "4 - principles of modern neuroimaging," in *Principles of Neurological Surgery (Fourth Edition)* (R. G. Ellenbogen, L. N. Sekhar, N. D. Kitchen, and H. B. da Silva, eds.), pp. 62–86.e2, Philadelphia : Elsevier, fourth edition ed., 2018.

[22] L. R. Ment, D. Scheinost, and T. Constable, "14 - microstructural and functional connectivity in the developing brain," in *Swaiman's Pediatric Neurology (Sixth Edition)* (K. F. Swaiman, S. Ashwal, D. M. Ferriero, N. F. Schor, R. S. Finkel, A. L. Gropman, P. L. Pearl, and M. I. Shevell, eds.), pp. 97–106, Elsevier, sixth edition ed., 2017.

[23] R. Dahnke and C. Gaser, "Computational anatomy toolbox flyer," 09 2019.

[24] M. Jenkinson, C. F. Beckmann, T. E. Behrens, M. W. Woolrich, and S. M. Smith, "Fsl," *NeuroImage*, vol. 62, no. 2, pp. 782–790, 2012. 20 YEARS OF fMRI.

[25] S. Whitfield-Gabrieli and A. Nieto-Castanon, "Conn : A Functional Connectivity Toolbox for Correlated and Anticorrelated Brain Networks," *Brain Connectivity*, vol. 2, pp. 125–141, jun 2012.

[26] M. Brett, I. Johnsrude, and A. Owen, "Opinionthe problem of functional localization in the human brain," *Nature reviews. Neuroscience*, vol. 3, pp. 243–9, 04 2002.

[27] A. Horn, "Mni t1 6thgen nlin to mni 2009b nlin ants transform," Jul 2016.

[28] S. Lorio, S. Fresard, S. Adaszewski, F. Kherif, R. Chowdhury, R. Frackowiak, J. Ashburner, G. Helms, N. Weiskopf, A. Lutti, and B. Draganski, "New tissue priors for improved automated classification of subcortical brain structures on mri," *NeuroImage*, vol. 130, pp. 157–166, 2016.

[29] L. Kalia and D. Lang, "Parkinson's disease," *The Lancet*, vol. 386, 04 2015.

[30] B. G. Buchanan and E. H. Shortliffe, eds., *Rule-Based Expert Systems : The MYCIN Experiments of the Stanford Heuristic Programming Project*. Reading, MA : Addison-Wesley, 1985.

[31] N. K. Focke, G. Helms, S. Scheewe, P. M. Pantel, C. G. Bachmann, P. Dechent, J. Ebentheuer, A. Mohr, W. Paulus, and C. Trenkwalder, "Individual voxel-based subtype prediction can differentiate progressive supranuclear palsy from idiopathic parkinson syndrome and healthy controls," *Human Brain Mapping*, vol. 32, no. 11, pp. 1905–1915, 2011.

[32] S. Esmaeilzadeh, Y. Yang, and E. Adeli, "End-to-end parkinson disease diagnosis using brain mr-images by 3d-cnn," 06 2018.

[33] H. Chen, Q. Dou, L. Yu, and P. Heng, "Voxresnet : Deep voxelwise residual networks for volumetric brain segmentation," *CoRR*, vol. abs/1608.05895, 2016.

[34] O. Ronneberger, P. Fischer, and T. Brox, "U-net : Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015.

[35] F. Milletari, N. Navab, and S. Ahmadi, "V-net : Fully convolutional neural networks for volumetric medical image segmentation," *CoRR*, vol. abs/1606.04797, 2016.

[36] C. Han, H. Hayashi, L. Rundo, R. Araki, W. Shimoda, S. Muramatsu, Y. Furukawa, G. Mauri, and H. Nakayama, "Gan-based synthetic brain mr image generation," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pp. 734–738, 2018.

[37] J. H. Cole, R. P. Poudel, D. Tsagkrasoulis, M. W. Caan, C. Steves, T. D. Spector, and G. Montana, "Predicting brain age with deep learning from raw imaging data results in a reliable and heritable biomarker," *NeuroImage*, vol. 163, pp. 115–124, 2017.

[38] F. Martínez-Murcia, A. Ortiz, J. Gorriz, J. Ramírez, F. Segovia, D. Salas-Gonzalez, D. Castillo-Barnes, and I. Illan, "A 3d convolutional neural network approach for the diagnosis of parkinson's disease," pp. 324–333, 05 2017.

[39] S. Sarraf, G. Tofighi, and for the Alzheimer's Disease Neuroimaging Initiative, "Deepad : Alzheimer's disease classification via deep convolutional neural networks using mri and fmri," *bioRxiv*, 2016.

[40] S. Korolev, A. Safiullin, M. Belyaev, and Y. Dodonova, "Residual and plain convolutional neural networks for 3d brain MRI classification," *CoRR*, vol. abs/1701.06643, 2017.

[41] E. Hosseini-Asl, G. L. Gimel'farb, and A. El-Baz, "Alzheimer's disease diagnostics by a deeply supervised adaptable 3d convolutional network," *CoRR*, vol. abs/1607.00556, 2016.

[42] B. Khagi, C. G. Lee, and G.-R. Kwon, "Alzheimer's disease classification from brain mri based on transfer learning from cnn," in *2018 11th Biomedical Engineering International Conference (BMEiCON)*, pp. 1–4, 2018.

[43] K. Thung and P.-T. Yap, "Multi-stage diagnosis of alzheimer's disease with incomplete multimodal data via multi-task deep learning," vol. 10553, pp. 160–168, 09 2017.

[44] S. Shinde, S. Prasad, Y. Saboo, R. Kaushick, J. Saini, P. K. Pal, and M. Ingalhalikar, "Predictive markers for parkinson's disease using deep neural nets on neuromelanin sensitive mri," *NeuroImage : Clinical*, vol. 22, p. 101748, 2019.

[45] S. Rajandran Nair, L. Tan, N. Ramli, S.-Y. Lim, K. Rahmat, and H. Nor, "A decision tree for differentiating multiple system atrophy from parkinson's disease using 3-t mr imaging," *European radiology*, vol. 23, 01 2013.

[46] F. Nemmi, A. P.-L. Traon, O. Phillips, M. Galitzky, W. Meissner, O. Rascol, and P. Peran, "A totally data-driven whole-brain multimodal pipeline for the discrimination of parkinson's disease, multiple system atrophy and healthy control," *NeuroImage : Clinical*, vol. 23, p. 101858, 2019.

[47] P. Peran, G. Barbagallo, F. Nemmi, M. Sierra, M. Galitzky, A. P.-L. Traon, P. Payoux, W. G. Meissner, and O. Rascol, "Mri supervised and unsupervised classification of parkinson's disease and multiple system atrophy," *Movement Disorders*, vol. 33, no. 4, pp. 600–608, 2018.

[48] E. Baudou, F. Nemmi, M. Biotteau, S. Maziero, P. Peran, and Y. Chaix, "Can the cognitive phenotype in neurofibromatosis type 1 (nf1) be explained by neuroimaging ? a review," *Frontiers in Neurology*, vol. 10, p. 1373, 2020.

[49] Y. Yoo, L. W. Tang, T. Brosch, D. K. B. Li, L. Metz, A. Traboulsee, and R. Tam, "Deep learning of brain lesion patterns for predicting future disease activity in patients with early symptoms of multiple sclerosis," in *Deep Learning and Data Labeling for Medical Applications* (G. Carneiro, D. Mateus, L. Peter, A. Bradley, J. M. R. S. Tavares, V. Belagiannis, J. P. Papa, J. C. Nascimento, M. Loog, Z. Lu, J. S. Cardoso, and J. Cornebise, eds.), (Cham), pp. 86–94, Springer International Publishing, 2016.

[50] P. Peran, B. Malagurski, F. Nemmi, B. Sarton, H. Vinour, F. Ferre, F. Bounes, D. Rousset, S. Mrozeck, T. Seguin, B. Riu, V. Minville, T. Geeraerts, J. A. Lotterie, X. Deboissezon, J. F. Albucher, O. Fourcade, J. M. Olivot, L. Naccache, and S. Silva, "Functional and Structural Integrity of Frontoparietal Connectivity in Traumatic and Anoxic Coma," *Critical Care Medicine*, vol. 48, no. 8, 2020.

[51] S. Vieira, W. Pinaya, and A. Mechelli, "Using deep learning to investigate the neuroimaging correlates of psychiatric and neurological disorders : Methods and applications," *Neuroscience & Biobehavioral Reviews*, vol. 74, 01 2017.

[52] J. Xin, Y. Zhang, Y. Tang, and Y. Yang, "Brain differences between men and women : Evidence from deep learning," *Frontiers in Neuroscience*, vol. 13, p. 185, 2019.

[53] C. Yang, A. Rangarajan, and S. Ranka, "Visual explanations from deep 3d convolutional neural networks for alzheimer's disease classification," *CoRR*, vol. abs/1803.02544, 2018.

[54] J. Kawahara, C. J. Brown, S. P. Miller, B. G. Booth, V. Chau, R. E. Grunau, J. G. Zwicker, and G. Hamarneh, "Brainnetcnn : Convolutional neural networks for brain networks ; towards predicting neurodevelopment," *NeuroImage*, vol. 146, pp. 1038–1049, 2017.

[55] E. Gibson, W. Li, C. Sudre, L. Fidon, D. I. Shakir, G. Wang, Z. Eaton-Rosen, R. Gray, T. Doel, Y. Hu, T. Whyntie, P. Nachev, M. Modat, D. C. Barratt, S. Ourselin, M. J. Cardoso, and T. Vercauteren, "Niftynet : a deep-learning platform for medical imaging," *Computer Methods and Programs in Biomedicine*, vol. 158, pp. 113–122, 2018.

[56] A. Fanciulli and G. K. Wenning, "Multiple-System Atrophy," *New England Journal of Medicine*, vol. 372, pp. 249–263, jan 2015.

[57] G. Barbagallo, M. Sierra-Peña, F. Nemmi, A. P.-L. Traon, W. G. Meissner, O. Rascol, and P. Peran, "Multimodal mri assessment of nigro-striatal pathway in multiple system atrophy and parkinson disease," *Movement Disorders*, vol. 31, no. 3, pp. 325–334, 2016.

[58] E. Adeli, F. Shi, L. An, C.-Y. Wee, G. Wu, and T. Wang, "Joint feature-sample selection and robust diagnosis of parkinson's disease from mri data," *NeuroImage*, vol. 141, 06 2016.

[59] F. D. Bowman, D. F. Drake, and D. E. Huddleston, "Multimodal imaging signatures of parkinson's disease," *Frontiers in Neuroscience*, vol. 10, p. 131, 2016.

[60] Y. Chen, W. Yang, J. Long, Y. Zhang, J. Feng, Y. Li, and B. Huang, "Discriminative analysis of parkinson's disease based on whole-brain functional connectivity," *PLOS ONE*, vol. 10, pp. 1–16, 04 2015.

[61] D. Long, J. Wang, M. Xuan, Q. Gu, X. Xu, D. Kong, and M. Zhang, "Automatic classification of early parkinson's disease with multi-modal mr imaging," *PLOS ONE*, vol. 7, pp. 1–9, 11 2012.

[62] B. Peng, S. Wang, Z. Zhou, Y. Liu, B. Tong, T. Zhang, and Y. Dai, "A multilevel-roi-features-based machine learning method for detection of morphometric biomarkers in parkinson's disease," *Neuroscience Letters*, vol. 651, pp. 88 – 94, 2017.

[63] D. Zhang, X. Liu, J. Chen, and B. Liu, "Distinguishing patients with parkinson's disease subtypes from normal controls based on functional network regional efficiencies," *PLOS ONE*, vol. 9, pp. 1–18, 12 2014.

[64] H.-J. Huppertz, L. Möller, M. Südmeyer, R. Hilker, E. Hattingen, K. Egger, F. Amtage, G. Respondek, M. Stamelou, A. Schnitzler, E. H. Pinkhardt, W. H. Oertel, S. Knake, J. Kassubek, and G. U. Höglinger, "Differentiation of neurodegenerative parkinsonian syndromes by volumetric magnetic resonance imaging analysis and support vector machine classification," *Movement Disorders*, vol. 31, no. 10, pp. 1506–1517, 2016.

[65] F. Nemmi, F. Cignetti, C. Assaiante, S. Maziero, F. Audic, P. Péran, and Y. Chaix, "Discriminating between neurofibromatosis-1 and typically developing children by means of multimodal MRI and multivariate analyses," *Human Brain Mapping*, May 2019.

[66] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," 01 2016.

[67] D. Perkins and G. Salomon, "Transfer of learning," vol. 11, 07 1999.

[68] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," 2020.

[69] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation : Representing model uncertainty in deep learning," 2016.

[70] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *Journal of Machine Learning Research - Proceedings Track*, vol. 9, pp. 249–256, 01 2010.

[71] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on Machine Learning* (S. Dasgupta and D. McAllester, eds.), vol. 28 of *Proceedings of Machine Learning Research*, (Atlanta, Georgia, USA), pp. 1139–1147, PMLR, 17–19 Jun 2013.

[72] D. Kingma and J. Ba, "Adam : A method for stochastic optimization," *International Conference on Learning Representations*, 12 2014.

[73] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of adam & beyond," 05 2018.

[74] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, p. 2121–2159, July 2011.

[75] M. D. Zeiler, "Adadelta : An adaptive learning rate method," 2012.

[76] D. P. Kingma and J. Ba, "Adam : A method for stochastic optimization," 2017.

[77] T. Dozat, "Incorporating nesterov momentum into adam," 2016.

[78] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks : Visualising image classification models and saliency maps," 2014.

[79] T. N. Mundhenk, B. Y. Chen, and G. Friedland, "Efficient saliency maps for explainable ai," 2020.

[80] B. Zhou, A. Khosla, L. A., A. Oliva, and A. Torralba, "Learning Deep Features for Discriminative Localization.," *CVPR*, 2016.

[81] R. Rs, M. Cogswell, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam : Visual explanations from deep networks via gradient-based localization," pp. 618–626, 10 2017.

[82] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-cam++ : Generalized gradient-based visual explanations for deep convolutional networks," *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Mar 2018.

[83] D. Omeiza, S. Speakman, C. Cintas, and K. Weldermariam, "Smooth grad-cam++ : An enhanced inference level visualization technique for deep convolutional neural network models," 2019.

[84] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "Smoothgrad : removing noise by adding noise," 2017.

[85] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010.

[86] E. Villain, G. M. Mattia, F. Nemmi, P. Péran, X. Franceries, and M. V. le Lann, "Visual interpretation of cnn decision-making process using simulated brain mri," in *2021 IEEE 34th International Symposium on Computer-Based Medical Systems (CBMS)*, pp. 515–520, 2021.

[87] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research),"

[88] G. M. Mattia, F. Nemmi, E. Villain, M.-V. Le Lann, X. Franceries, and P. Péran, "Investigating the discrimination ability of 3d convolutional neural networks applied to altered brain mri parametric maps," Jul 2021.

[89] W. G. Meissner, P.-O. Fernagut, B. Dehay, P. Péran, A. P.-L. Traon, A. Foubert-Samier, M. Lopez Cuina, E. Bezard, F. Tison, and O. Rascol, "Multiple system atrophy : Recent developments and future perspectives," *Movement Disorders*, vol. 34, no. 11, pp. 1629–1642, 2019.

[90] C. Scherfler, G. Göbel, C. Müller, M. Nocker, G. K. Wenning, M. Schocke, W. Poewe, and K. Seppi, "Diagnostic potential of automated subcortical volume segmentation in atypical parkinsonism," *Neurology*, vol. 86, p. 1242—1249, March 2016.

[91] C. G. Goetz, W. Poewe, O. Rascol, C. Sampaio, G. T. Stebbins, C. Counsell, N. Giladi, R. G. Holloway, C. G. Moore, G. K. Wenning, M. D. Yahr, and L. Seidl, "Movement disorder society task force report on the hoehn and yahr staging scale : Status and recommendations the movement disorder society task force on rating scales for parkinson's disease," *Movement Disorders*, vol. 19, no. 9, pp. 1020–1028, 2004.

[92] Y. J. Zhao, H. L. Wee, Y.-H. Chan, S. H. Seah, W. L. Au, P. N. Lau, E. C. Pica, S. C. Li, N. Luo, and L. C. Tan, "Progression of parkinson's disease as evaluated by hoehn and yahr stage transition times," *Movement Disorders*, vol. 25, no. 6, pp. 710–716, 2010.

[93] C. Gaser and R. Dahnke, "CAT - A Computational Anatomy Toolbox for the Analysis of Structural MRI Data," in *HBM*, 2016.

[94] T. Behrens, M. Woolrich, M. Jenkinson, H. Johansen-Berg, R. Nunes, S. Clare, P. Matthews, J. Brady, and S. Smith, "Characterization and propagation of uncertainty in diffusion-weighted MR imaging," *Magnetic Resonance in Medicine*, vol. 50, pp. 1077–1088, nov 2003.

[95] Q.-H. Zou, C.-Z. Zhu, Y. Yang, X.-N. Zuo, X.-Y. Long, Q.-J. Cao, Y.-F. Wang, and Y.-F. Zang, "An improved approach to detection of amplitude of low-frequency fluctuation (ALFF) for resting-state fMRI : fractional ALFF.," *Journal of neuroscience methods*, vol. 172, pp. 137–41, jul 2008.

[96] S. Ioffe and C. Szegedy, "Batch normalization : Accelerating deep network training by reducing internal covariate shift," 02 2015.

[97] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[98] W. Boulila, H. Ghandorh, M. A. Khan, F. Ahmed, and J. Ahmad, "A novel cnn-lstm-based approach to predict urban expansion," 2021.

[99] N. T. Nguyen, D. Q. Tran, N. T. Nguyen, and H. Q. Nguyen, "A cnn-lstm architecture for detection of intracranial hemorrhage on ct scans," 2020.